New Techniques for Probabilistically Checkable Proofs and Inapproximability Results

Subhash Khot

A Dissertation
Presented to the Faculty
of Princeton University
in Candidacy for the Degree
of Doctor of Philosophy

Recommended for Acceptance
By the Department of
Computer Science

November 2003

Abstract

Certain NP-hard problems like Clique and MAX-3SAT have resisted all attempts to find non-trivial approximation. Is there any inherent reason for the apparent inapproximability of these problems ? The discovery of PCP Theorem and subsequent research have shown that for Clique and MAX-3SAT, any non-trivial approximation is as hard as finding the exact solution !

In this work, we continue this line of research and show inapproximability of many fundamental NP-hard problems. These include (Hyper-)Graph Coloring, Shortest Vector Problem (SVP) in lattices, Hypergraph Vertex Cover, Clique and Chromatic Number of graphs, and some results based on the Unique Games Conjecture (UGC) that we propose. Specifically, we show that :

(Hyper-)Graph Coloring : It is hard to color (i) k-colorable graphs with $k^{\Omega(\log k)}$ colors, (ii) 3-colorable 3-uniform hypergraphs with $(\log \log n)^{\Omega(1)}$ colors, and (iii) k-colorable 4-uniform hypergraphs with $(\log n)^{\Omega(k)}$ colors.

SVP : For all large enough p, it is hard to find the shortest nonzero vector in an n-dimensional lattice under L_p norm within factor $p^{1-o(1)}$.

Hypergraph Vertex Cover : The vertex cover in k-uniform hypergraphs is hard to approximate within factor $k-1-o(1)$ for every $k \geq 3$.

Clique and Chromatic Number of Graphs : Both these problems are hard to approximate within factor $\frac{n}{2^{(\log n)^{\gamma}}}$ for some constant $\gamma < 1$.

Consequences of UGC : We propose a conjecture about certain 2-Prover-1-Round games and show that it implies any constant factor hardness for Min-2SAT-Deletion and factor $k - o(1)$ hardness for vertex cover in k-uniform hypergraphs for every $k \geq 2$.

We use the powerful machinery of Probabilistically Checkable Proofs and introduce many new techniques for constructing and analyzing PCPs.

Acknowledgments

I am forever indebted to my advisor Sanjeev Arora for shaping every aspect of my academic life. He has always helped me make right decisions : to join Princeton, to pick my research area, and finally to leave gradschool and move on. He made persistent efforts to improve my writing and presentation skills, and I am to blame for his failure, if any.

I am most thankful to Andy Yao, Johan Håstad and Venkatesan Guruswami for guiding my research in correct direction. Andy got me interested in complexity; he is the best teacher I have seen. Johan and Venkat have been a constant source of new problems, ideas, and encouragement. Most of my work was done when Johan was at IAS and the year after when Venkat was at Berkeley.

I am grateful to everybody who has contributed to the awesome research environment at Princeton : faculty, graduate students, and visitors at Princeton, IAS, NEC, Rutgers, and DIMACS. Thanks to everybody in Berkeley for the wonderful year I spent there and everybody at IBM, Almaden for an enjoyable summer.

My research wouldn't have been possible but for my co-authors : Sanjeev, Johan, Venkat, Irit Dinur, Oded Regev, Jonas Holmerin, Amit Chakrabarti, Yaoyun Shi, Ravi Kumar, Jayram Thathachar, Yuval Rabani, Xiaodong Sun, and Venkatesh Raman. I thank them all for patiently listening to my random ideas and suggesting ideas that actually work.

Many more people have directly or indirectly influenced my research. At the risk of leaving out many of them, I acknowledge Bernard Chazelle, Amit Sahai, Moses Charikar, D. Sivakumar, Ziv Bar-Yosef, S. Venkatesh, Satya Lokam, Ding Liu, Manoj Prabhakaran, Kunal Talwar, Robi Krauthgamer, and Eran Halperin. I owe a great deal to those who have instilled in me love for mathematics and problem solving. These include my undergraduate advisor Sundar Vishwanathan and faculty members at IIT-Bombay, IMSc, and the IMO Training Program.

I would like to take this opportunity to thank all my friends, who met me at different junctures of my life and became an inseparable part of it. Words would fall short to thank my teacher Gogate Sir; he is responsible for moulding my highschool years and continues to provide guidance and inspiration. Finally, my mom and my brother Amol, they mean everything to me ...

Subhash Khot

I gratefully acknowledge Sanjeev Arora's NSF Awards 0205594, CCR 0098180 and David and Lucile Packard Fellowship. I also acknowledge Wu Fellowship of Princeton University and IBM PhD Fellowship.

To the memories of my beloved father ...

Contents

Abstract iii

1 Introduction 1

1.1 Approximability of NP-hard Problems 1

1.2 Work in this Thesis 4

1.2.1 Organization of Thesis 10

1.3 The PCP Theorem 10

1.4 Research Leading to PCP Theorem 15

1.5 Progress after PCP Theorem 15

2 A Framework for Building PCPs 20

2.1 Plan for Building 3-Bit PCP 22

2.2 The 2-Prover 1-Round Game 24

2.3 The Label Cover Problem 27

2.4 Long Codes and Fourier Analysis 28

2.5 Håstad's 3-Bit PCP 31

2.5.1 Completeness 33

2.5.2 Soundness 34

2.6 A General Approach to PCP Design 37

2.7 Techniques in this Thesis 40

3 A Query Efficient PCP and Hardness of Graph Coloring 43

3.1 Definitions, Results and Techniques 44

3.2 Basic test 49

3.3 The Iterated Test (Almost Disjoint Sets Test) 54

3.4 Hardness of Graph Coloring 58

3.4.1 The FGLSS Graph 59

3.4.2 Randomized Label Cover 61

3.4.3 Pruning FGLSS Graph and Proof of Theorem 3.1.1 62

4 Hardness of Coloring 3-Uniform Hypergraphs 65

4.1 Definitions, Results and Techniques 66

4.2 Constructing Multi-layered Smooth Label Cover 73

4.2.1 Achieving Smoothness Property 73

4.2.2 Definition of Multi-layered Label Cover Problem 75

4.2.3 The Main Construction 78

4.3 Long Codes over $\mathbf{Z}_k$ and Fourier Analysis 81

4.4 Hardness of 3-Uniform Hypergraph Coloring 83

4.4.1 Soundness of PCP 84

4.5 Hardness of the Problem NAE$_{3,k}$ 89

4.5.1 Soundness of PCP 91

5 Hardness of Coloring 4-Uniform Hypergraphs 96

5.1 Results and Techniques 97

5.2 Preliminaries 99

5.2.1 Split Label Cover Problem . . . 99

5.2.2 The Split Code . . . 100

5.2.3 Fourier Analysis of Split Codes . . . 101

5.2.4 Equality Folding of Split Codes . . . 102

5.3 The PCP Construction . . . 104

5.4 Soundness Analysis . . . 107

5.5 Proofs of Lemmas 5.4.3, 5.4.4 and Theorem 5.1.1 . . . 112

6 Hardness of Shortest Vector Problem in High L_p Norms 115

6.1 Result and History of the Problem . . . 116

6.2 Problem Definition and Techniques . . . 119

6.3 The Basic Idea in the Reduction . . . 123

6.4 Full Reduction . . . 126

6.5 Soundness of the Reduction . . . 129

7 Hardness of Vertex Cover in k-Uniform Hypergraphs 135

7.1 Results and Techniques . . . 136

7.2 Biased Long Code and Intersecting Families . . . 142

7.3 The Hypergraph Construction . . . 144

7.4 Soundness of the Construction . . . 147

8 Unique Games Conjecture and its Consequences 151

8.1 Conjecture, its Motivation and Results . . . 152

8.2 Unique Label Cover Problem . . . 155

8.3 Hardness of Max-2-Lin-2 . . . 157

8.3.1 Bourgain's Theorem and Soundness Analysis . . . 159

8.4 Proof of Theorem 8.1.2 163

8.5 Conclusion 170

9 Hardness of Vertex Cover Based on Unique Games Conjecture 172

9.1 Techniques 173

9.2 Constructing the Strong Label Cover 174

9.3 Tools from Sensitivity Analysis of Boolean Functions 180

9.3.1 Friedgut's 'Core' Theorem 181

9.4 Reduction to Vertex Cover in k-Uniform Hypergraphs 184

9.4.1 Construction of the Hypergraph 185

9.4.2 Soundness 187

9.4.3 Proof of Lemma 9.4.1 189

10 Hardness of Clique and Chromatic Number 193

10.1 Results and Techniques 194

10.2 Preliminaries 198

10.2.1 The Raz Verifier 199

10.2.2 Fourier Analysis 200

10.2.3 Hadamard Codes, their Decoding and Folding 201

10.3 The Main PCP Construction and Analysis 203

10.4 Improved Inapproximability Result for MaxClique 210

10.5 Randomized PCPs and Chromatic Number 212

10.6 Randomized PCP for Coverable Max-3-Lin-$2(\epsilon)$ 215

10.6.1 Improved Inapproximability Result for Chromatic Number 217

10.7 Proof of Theorem 10.5.4 218

11 Conclusion **222**

11.1 Open Problems 222

11.2 Future Directions 224

11.2.1 Unique Games Conjecture 225

11.2.2 Feige's Hypothesis about Random 3SAT 225

List of Figures

1.1 PCP Verifier 12

2.1 Overview of Håstad's 3-Bit PCP 22

2.2 Label Cover, Long Code and 3-Bit PCP 32

3.1 FGLSS Graph 60

4.1 Multi-layered Label Cover 75

5.1 PCP for 4-Uniform Hypergraph Coloring 105

Chapter 1

Introduction

A central theme in complexity theory is lower bounds, i.e. results showing that certain computational problems are provably hard to solve. Many such results are based on complexity theoretic assumptions like P $\neq$ NP due to absence of unconditional lower bounds in general computational models like Turing machine. This work focuses on one specific class of lower bounds : we show that computing approximate solutions to certain NP-complete problems is hard.

1.1 Approximability of NP-hard Problems

Many optimization problems of theoretical and practical interest are NP-hard to solve exactly. Designing approximation algorithms is a popular and extensively studied approach for dealing with NP-completeness. An approximation algorithm with ratio $C > 1$ is a polynomial time (possibly randomized) algorithm that computes a solution $A(I)$ for every problem instance I such that (i) $A(I) \leq C \cdot OPT(I)$ for minimization problems and (ii) $A(I) \geq \frac{OPT(I)}{C}$ for maximization problems. Here $OPT(I)$ denotes the value of the optimum solution to the problem instance. We refer to a recent book of Vazirani [113] for an excellent treatment of the field of approximation algorithms.

It turns out that different NP-hard problems behave very differently in terms of how well they can be approximated. Let us look at three fundamental problems : MAX-3SAT, Bin-Packing and Set Cover.

MAX-3SAT : Given a set of n boolean variables and a set of m 3CNF clauses (i.e. clauses of type $x \vee \overline{y} \vee z$), find a truth assignment to variables satisfying maximum number of clauses. If one assigns TRUE or FALSE randomly and independently to every variable, every clause is satisfied with probability $\frac{7}{8}$. Thus a random assignment satisfies $\frac{7}{8}$ fraction of the clauses in expected sense and it is easy to convert this to a deterministic algorithm as well. Thus we get a trivial approximation algorithm with ratio $\frac{8}{7}$.

Bin-Packing : Given a set of integers $\{x_i\}_{i=1}^n$ and a *bin-size* L, partition the set into minimum number of sets $S_1, S_2, \dots, S_t$ such that $\sum_{x_i \in S_l} x_i \leq L$ for $1 \leq l \leq t$. Bin-packing has an approximation algorithm with ratio $1 + \epsilon$ running in time $n^{O(1/\epsilon^2)}$ for every $\epsilon > 0$ (see [113, Chapter 9]). We say that Bin-Packing has a *Polynomial Time Approximation Scheme* or PTAS, meaning $1 + \epsilon$ approximation for every ϵ. The running time of the algorithm depends on ϵ, but for every fixed ϵ, the running time is polynomial.

Set Cover : Given a ground set X with $|X| = n$ and a collection of subsets $S_1, S_2, \dots S_m \subseteq X$ with $\cup_{i=1}^m S_i = X$. The goal is to find a minimum size sub-collection $S_{i_1}, S_{i_2}, \dots, S_{i_t}$ such that $\cup_{j=1}^t S_{i_j} = X$. A greedy algorithm with approximation ratio $\ln n$ is known, often taught in a undergraduate class (see [113, Chapter 2]).

Thus MAX-3SAT has a constant factor approximation, Bin-Packing in fact has a PTAS and for Set Cover the approximation ratio grows as a function of the input size. In particular, we don't know of any constant factor approximation for Set Cover. Is there any inherent reason why these problems behave differently ? Is there a fundamental limitation to designing better algorithms for MAX-3SAT and Set Cover or just that we haven't been clever enough to find the right algorithm ?

This was a natural and perplexing question, unresolved till early 90's. Surprisingly, the answer came via algebraic techniques developed in the study of interactive proofs. The celebrated PCP Theorem (by Arora, Safra [13] and Arora, Lund, Motwani, Sudan, Szegedy [12]) states that MAX-3SAT has no PTAS unless P = NP. In other words, MAX-3SAT (and a bunch of other problems) has a threshold $1 + \epsilon_0$ such that approximating the problem better than this threshold is as hard as solving it exactly ! Such results that rule out the possibility of a good approximation algorithm are called *inapproximability results* or *hardness results*. The PCP Theorem has had tremendous impact on the theory of inapproximability. For instance, subsequent work has shown that for MAX-3SAT and Set Cover, the simple algorithms mentioned before are in fact optimal ! So $\frac{8}{7}$ and $\ln n$ are tight *approximability thresholds* for the respective problems.

PCP Theorem can be equivalently stated as a result on proof checking. The theorem gives a way of specifying proofs for NP-statements such that the proofs can be *spot-checked* very efficiently by a probabilistic verifier. The verifier uses a minimal amount of randomness and reads only a constant number of bits from the proof ! The discovery of the connection between proof checking and inapproximability results (by Feige, Goldwasser, Lovász, Safra and Szegedy [38]) is one of the most exciting theoretical developments in the last decade. It demonstrates how complexity theoretic tools can be used to answer questions arising from algorithm design.

1.2 Work in this Thesis

After the discovery of PCP Theorem, much research has focused on obtaining tight inapproximability thresholds for NP-hard problems. The hardness results for MAX-3SAT and Set Cover are excellent examples of the success of PCP techniques. In this thesis, we continue this line of research and explore hardness of basic problems like Graph Coloring, Vertex Cover, Shortest Vector Problem, Clique and Chromatic Number.

(Hyper-)Graph Coloring

A graph is called k-colorable if one can assign one color to every vertex of the graph using at most k colors, such that every edge has its endpoints colored with different colors. It is well-known that 3-Colorability of graphs is a NP-hard problem. Given a 3-colorable graph, how many colors does one need to find a valid coloring in polynomial time ? This question is of great interest in combinatorics and computer science.

The best known algorithm ([70], [18]) colors 3-colorable graphs with $\tilde{O}(n^{3/14})$ colors. The number of colors used is huge (polynomial in number of vertices), and even such a modest guarantee relies on sophisticated combinatorics and semidefinite programming. The hardness results are not satisfactory either : all we know is it is NP-hard to color 3-colorable graphs with 4 colors [72] and for large enough k, it is hard to color k-colorable graphs with $k^{3/2}$ colors [47]. In this thesis, we make a remarkable progress on this problem. We show that it is NP-hard to color k-colorable graphs with $k^{\Omega(\log k)}$ colors; the first superpolynomial bound for this problem as a function of k (Theorem 3.1.1). The result is based on a PCP construction where the verifier is highly query-efficient and has perfect completeness. The goal of this line of work would be to show NP-hardness of coloring

3-colorable graphs with any constant number of colors; such a result however seems out of reach of the current techniques.

Given this state of affairs, it is natural to ask whether strong hardness results can be shown for coloring hypergraphs. In a q-uniform hypergraph, every edge is a size-q subset of vertices (thus graphs are 2-uniform hypergraphs). A hypergraph is called k-colorable if its vertices can be colored with at most k colors so that every edge is non-monochromatic, i.e. for every edge, not all its vertices get the same color. For q-uniform hypergraphs with $q \geq 3$, even 2-Colorability is NP-hard. For $q \geq 3, k \geq 2$, algorithms for coloring k-colorable q-uniform hypergraphs with $n^{\Omega(1)}$ colors are known (e.g. [81]). It came as a surprise when Guruswami, Håstad, Sudan [55] were able to show that it is hard to color 2-colorable 4-uniform hypergraphs with $\Omega(\frac{\log\log n}{\log\log\log n})$ colors. For k-colorable 4-uniform hypergraphs, we improve this bound to $(\log n)^{\Omega(k)}$ colors which exceeds every poly($\log n$) function as k grows (Theorem 5.1.1). This gives an evidence that for (hyper)graph coloring, the right answer might be super-polylogarithmic.

What happens for 3-uniform hypergraphs ? Guruswami et al's techniques do not extend to 3-uniform hypergraphs. We introduce a new technique called *Multi-layered Smooth Label Cover* (Theorem 4.2.4) and settle the 3-uniform case. We show that it is hard to color 3-colorable 3-uniform hypergraphs with $(\log\log n)^{\Omega(1)}$ colors (Theorem 4.1.1). We also obtain new results for a variation of coloring problem where one is required to use a fixed number of colors and maximize the number of non-monochromatic edges. This question has been resolved for graphs and 4-uniform hypergraphs whereas we settle the 3-uniform case (Theorem 4.1.3).

Shortest Vector Problem

An n-dimensional lattice $\mathcal{L}$ is a set of vectors $\{\sum_{i=1}^{n} a_i v_i \mid a_i \in \mathbf{Z}\}$ where $v_1, v_2, \ldots, v_n \in \mathbf{R}^m$ is a set of independent vectors called the basis for the lattice. The same lattice could have many bases. Given a basis for an n-dimensional lattice, the Shortest Vector Problem asks for the shortest non-zero vector in the lattice. The length of the vectors can be measured in any L_p norm ($p \geq 1$) and the corresponding problem is denoted by SVP$_p$.

SVP is one of the most beautiful problems with connections to worst-case to average-case reduction, breaking and building cryptosystems (!), factoring rational polynomials and numerous other areas in mathematics and computer science. We refer to Chapter 6 for the history and significance of this problem. A polynomial time algorithm achieving $2^{o(n)}$-approximation to SVP is known (the famous LLL Algorithm [87] and improvement by [107]). This is a rather weak approximation guarantee and it is a major open question whether a polynomial factor approximation exists. SVP$_\infty$ is long since known to be NP-hard whereas for $p < \infty$, even NP-hardness wasn't known until recently. In a breakthrough result, Ajtai [2] showed NP-hardness of SVP$_2$ and then Micciancio [96] showed factor $2^{1/p} - o(1)$ inapproximability result for SVP$_p$. Both these results use a randomized reduction and hold under assumption NP $\not\subseteq$ BPP.

In this thesis, we show factor $p^{1-o(1)}$ hardness for SVP$_p$ for all sufficiently large values of p (Theorem 6.1.1). Apart from the improved hardness factor, our reduction is much simpler and direct, much more *elementary*, and holds under the weaker assumption NP $\not\subseteq$ ZPP. We believe that our ideas could be applicable in getting any constant factor hardness in some fixed L_p (maybe even in L_2) norm.

Hypergraph Vertex Cover

Vertex Cover in a graph is a set of vertices that touches every edge in the graph. Finding minimum vertex cover is a basic NP-hard problem. A greedy algorithm with approximation ratio 2 is well-known and in spite of a great deal of efforts, no better approximation is known. The problem is known to have *integrality gap* of 2 for a wide class of linear programs (see [10]) suggesting that LP-based approach is unlikely to give a better approximation. It is believed (though some might disagree) that factor 2 is indeed optimal. Showing $2 - o(1)$ hardness for vertex cover is equivalent to a fundamental open question regarding PCPs with *zero free bits* and seems to be a first step towards Graph Coloring problem. The best hardness known is 1.36 due to Dinur and Safra [32] and the current techniques seems to have stuck on this important problem.

Therefore it is natural to investigate the generalization of vertex cover to k-uniform hypergraphs with $k \geq 3$. A factor k approximation algorithm follows easily and no better algorithm is known. Trevisan [111] showed $k^{1/19}$ hardness for this problem and Holmerin ([66], about the same time as our work) showed $k^{1-o(1)}$ hardness. In this thesis, we show that vertex cover in k-uniform hypergraphs is hard to approximate within factor $k - 1 - o(1)$ for every $k \geq 3$ (Theorem 7.1.1). The result is almost optimal.

Our reduction introduces a new technique called Multi-layered Label Cover (Theorem 4.2.4), also used in proving hardness of 3-uniform hypergraph coloring. Vertex Cover on k-uniform hypergraphs can be viewed as a special case of Set Cover. Our reduction has an important feature not shared by Set Cover reductions known earlier, viz. in the *bad case*, almost all sets are required to cover the universe. Our result has subsequently been used in showing optimal $\Omega(\log^* n)$ hardness for Asymmetric k-Center Problem [26].

Unique Games Conjecture and its Consequences

Though we make significant progress on Coloring and Vertex Cover for hypergraphs, we seem to be stuck on these problems for graphs. Towards the end of this thesis, we set out to investigate the limitations of current techniques and possible directions for future research.

We propose a conjecture about certain 2-Prover-1-Round games (Conjecture 8.1.1) and show that it implies any constant factor hardness for Min-2SAT-Deletion and factor $k - o(1)$ hardness for vertex cover in k-uniform hypergraphs for every $k \geq 2$ (Theorems 8.3.1 and 9.0.2 respectively). In particular, the conjecture would settle the Vertex Cover problem on graphs and therefore, also take an important step towards Graph Coloring problem. The conjecture can be motivated as follows. We are given a system of linear equations mod p where p is a constant, every equation contains two variables and there exists an assignment that satisfies almost all equations. Is it possible to find, in polynomial time, an assignment that satisfies δ fraction of equations where δ is a constant independent of p ? Intuitively, the answer should be NO and roughly speaking, that is what the conjecture is all about.

A typical PCP verifier is constructed by composition of two verifiers, an Outer Verifier and an Inner Verifier. The outer verifier is essentially a 2P1R game. The Unique Games Conjecture states existence of a new outer verifier. It can be used in conjunction with an appropriate inner verifier to prove desired hardness results.

We also present a semidefinite programming based algorithm (Theorem 8.1.2) that sheds some light on the truth of this conjecture. We believe that research directed towards resolving our conjecture would be very fruitful.

Clique and Chromatic Number of Graphs

A clique in a graph is a set of vertices such that all of them are pairwise connected by an edge. Finding maximum clique size in a graph is a fundamental NP-hard problem. An approximation algorithm with ratio n is trivial : just output one vertex as a clique. The best known algorithm does only slightly better, achieving a ratio of $O(\frac{n}{\log^2 n})$ [21]. A polynomial time computable function called Lovász θ-function was conjectured to give $\sqrt{n}$ approximation to MaxClique. This was disproved by Feige [36] who showed that θ-function has approximation ratio as bad as $\frac{n}{2^{O(\sqrt{\log n})}}$. In a breakthrough result, Håstad [59] in fact showed that for any $\epsilon > 0$, approximating MaxClique within factor $n^{1-\epsilon}$ is hard unless NP = ZPP. Hardness results for MaxClique are equivalent to constructing PCPs with so-called *low amortized free bit complexity* (see Theorem 3.1.4). Roughly speaking, this parameter measures the trade-off between number of queries and error probability of a PCP verifier. Håstad was able to construct a verifier with arbitrarily low amortized free bit complexity and this implies $n^{1-\epsilon}$ hardness for MaxClique.

Even after $n^{1-\epsilon}$ hardness result, it is still interesting to pin-point the exact hardness of MaxClique. Specifically, is $\frac{n}{2^{O(\sqrt{\log n})}}$ the right answer ? This question was also raised by Srinivasan [109]. Trevisan [111] showed factor $\frac{k}{2^{O(\sqrt{\log k})}}$ hardness for MaxClique on degree k-graphs (k thought of as a constant). Is it possible to interpolate this result all the way upto $\frac{n}{2^{O(\sqrt{\log n})}}$? In this thesis, we take a step towards resolving this question. We show a hardness factor of $\frac{n}{2^{(\log n)^\gamma}}$ for some $\gamma < 1$ (Theorem 10.1.2). Previously, Engebretsen and Holmerin [33] showed a hardness factor $\frac{n}{2^{O(\log n/\sqrt{\log\log n})}}$.

Our result is proved via a new PCP construction based on Hadamard Code (as opposed to popular long code). Hadamard codes are much shorter in length and yield proofs of much smaller size. We then apply a technique called *Randomized PCP* (Section 10.5) and show $\frac{n}{2^{(\log n)^\gamma}}$ hardness for approximating the Chromatic Number of a graph. Earlier

Feige, Kilian [40] and Engebretsen, Holmerin [33] showed hardness factor of $n^{1-\epsilon}$ and $\frac{n}{2^{O(\log n/\sqrt{\log\log n})}}$ respectively. They obtain randomized versions of PCPs based on long code. Our construction is much simpler.

Our results for clique and chromatic number differ in notation from the rest of the thesis and therefore, we present them towards the end of the thesis.

1.2.1 Organization of Thesis

In Chapter 2, we present the basic tools and definitions used in constructing probabilistic proof systems (PCPs). We also present a PCP construction due to Håstad which serves as a prototype for many of the constructions in this thesis. Each of the results mentioned above appears as a separate chapter. Each chapter can be read more or less independently, using certain results from previous chapters as a black-box. In the beginning of every chapter, we state the problem definition, previous results, and high level techniques. In Chapter 11, we conclude with a list of open problems and directions for future research.

1.3 The PCP Theorem

We give a formal statement of the PCP Theorem in this section. The next section gives a brief history of work on interactive proofs and program checking that culminated in this beautiful result.

A common and very fruitful approach in complexity theory is to identify a class of problems and show that they are equivalent in terms of their computational complexity. An excellent example is the theory of NP-completeness where one shows that all NP-complete problems are equivalent in terms of polynomial time computability. In inapproximability theory, first such result was obtained by Papadimitriou and Yannakakis

[101]. They identified a class called MAX-SNP; many natural problems like Vertex Cover, MAX-3SAT and MAX-CUT are complete problems for this class. They showed that either all MAX-SNP-complete problems have PTAS (i.e. $1+\epsilon$ approximation for every $\epsilon > 0$) or none of them does. Thus as far as the existence of PTAS is concerned, all MAX-SNP-complete problems are equivalent. However, the question whether MAX-SNP-complete problems have PTAS's was left open. The question was resolved by the PCP Theorem which showed that MAX-3SAT and therefore every MAX-SNP-complete problem has no PTAS unless P = NP. This result, proved by Arora, Safra [13] and Arora, Lund, Motwani, Sudan and Szegedy [12] can be stated as :

Theorem 1.3.1 *(PCP Theorem, [13], [12]) There is a polynomial time reduction from SAT (and hence from any NP problem) to MAX-3SAT mapping an instance ϕ of SAT to an instance ψ of MAX-3SAT such that :*

- *If ϕ is satisfiable, so is ψ, i.e. $OPT(\psi) = 1$.*
- *If ϕ is not satisfiable, then no assignment to ψ satisfies more than a fraction s of the clauses, i.e. $OPT(\psi) \leq s$.*

Here $OPT(\psi)$ is the maximum fraction of clauses that can be satisfied by any assignment and $s < 1$ is an absolute constant. In particular, it is NP-hard to approximate MAX-3SAT within factor $\frac{1}{s}$.

Thus it is possible to transform 3SAT formulae so that satisfiable formulae remain satisfiable and unsatisfiable ones become *highly unsatisfiable*. In other words, an error in the formula, if present, spreads everywhere (giving a hint that error correcting codes are used to prove the theorem). As indicated before, PCP Theorem can be stated equivalently as a result on proof checking. Before we do that, let us look at the classic definition of NP.

Definition 1.3.2 NP *is defined to be the class of languages L which have the following kind of proof system : There is a deterministic, polynomial-time verifier V with access to input x and a string Π which is supposedly a proof showing that $x \in L$. The length of Π is polynomially bounded in the length of x. The verifier reads the proof, performs a polynomial time checking procedure and accepts or rejects. The proof system has these properties :*

$$\begin{array}{llll} \textit{(Completeness :)} & x \in L & \Longrightarrow & \exists \ proof \ \ \Pi \ \textit{ such that } \quad V(\Pi) = \textit{accept} \\ \textit{(Soundness :)} & x \notin L & \Longrightarrow & \forall \ proofs \, \Pi, \qquad\qquad V(\Pi) = \textit{reject} \end{array}$$

In other words, $x \in L$ if and only if there exists a proof that the verifier accepts.

The PCP Theorem gives a new characterization of NP as the class of languages that have a proof system where the verifier is probabilistic and allowed to read only a few bits from the proof (instead of reading the whole proof). Let us first give a general definition of such a class of languages (see Fig. 1.1).

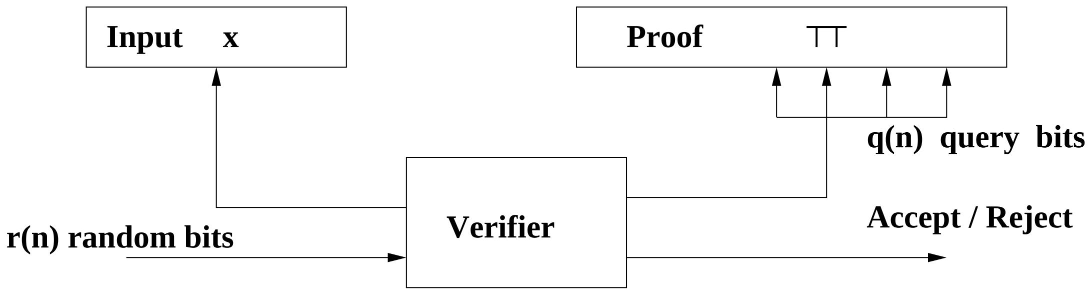


Figure 1.1: PCP Verifier

Definition 1.3.3 *Let $\mathrm{PCP}_{c,s}[r(n), q(n)]$ be the class of languages L which have the following kind of proof system : The verifier V is probabilistic and uses only $r(n)$ random bits where $n = |x|$ is the size of input. The verifier has access to a proof Π. Depending on* *the choice of its random coins, the verifier reads only $q(n)$ bits from the proof and then accepts or rejects. It satisfies :*

$$\begin{array}{llcll} (Completeness :) & x \in L & \Longrightarrow & \exists\, proof\ \Pi\ such\ that & \Pr[V \text{ accepts } \Pi] \geq c \\ (Soundness) : & x \notin L & \Longrightarrow & \forall\ proofs\,\Pi, & \Pr[V \text{ accepts } \Pi] \leq s \end{array}$$

$1 \geq c > s > 0$ *are called the completeness and soundness parameters respectively. If* $c = 1$*, the proof system is said to have perfect completeness.*

Remark : It is implicit in the definition that the running time of the verifier and size of the proof is at most $\max(2^{O(r(n))}, \mathrm{poly}(n))$. Since the verifier has only $2^{r(n)}$ different "runs" and reads only $q(n)$ bits in each run, the number of distinct proof locations accessed is at most $q(n)2^{r(n)}$. Thus the bound on the proof size makes sense. Also, since there are only $2^{r(n)}$ different runs, it is reasonable to demand that the verifier spends $2^{O(r(n))}$ time to figure out which bits to read and what his acceptance criterion should be for each run.

The PCP Theorem can now be stated as :

Theorem 1.3.4 *(PCP Theorem)*

$$\mathrm{NP} = \mathrm{PCP}_{1,\frac{1}{2}}[O(\log n), O(1)]$$

The verifier is polynomial time and the proof is of polynomial size.

Thus PCP Theorem states that NP-statements have proofs that can be checked using only logarithmic randomness and reading only a constant number of bits from the proof ! For a correct statement, there exists a proof that is always accepted. For a false statement, no proof is accepted with probability more that $\frac{1}{2}$. Thus any proof of a false statement must be wrong almost everywhere and reading a few bits suffices to catch such cheating proofs. Theorem 1.3.4 appears quite counter-intuitive and mystical. However, it is easy to see that it follows from Theorem 1.3.1. Assuming Theorem 1.3.1, we will show that every language in NP has a probabilistic proof system with appropriate parameters. Let $L \in \mathrm{NP}$. The PCP verifier proceeds as follows :

1. Given input x, transform it in poly-time to a SAT formula ϕ. This is possible since SAT is NP-complete.

2. Using the *magic reduction* from Theorem 1.3.1, transform ϕ to an instance ψ of MAX-3SAT.

3. Expect as a proof, an assignment to variables in ψ.

4. Check the proof as below :

   - Pick a clause (say $x \vee y \vee z$) at random from the set of all clauses in ψ.
   - Read the values of variables x, y, z from the proof.
   - Accept if and only if the values of x, y, z satisfy the clause.

Now let us verify that this proof system has the right parameters. If $n = |x|$, then $|\psi| =$ poly(n) and hence the verifier needs only $O(\log n)$ random bits to pick a random clause. The verifier reads only 3 bits from the proof which is a constant. For completeness condition, note that if $x \in L$, then ϕ is satisfiable and so is ψ. If we take a satisfying assignment to ψ as a proof, then the verifier accepts with probability 1. For soundness condition, note that if $x \notin L$, then ϕ is not satisfiable. Hence no assignment to ψ satisfies more than a fraction s of the clauses and consequently, no proof makes the verifier accept with probability more than s. The soundness parameter s can be brought down to $\frac{1}{2}$ by running the checking procedure a constant number of times.

The other direction (i.e. Theorem 1.3.4 implies Theorem 1.3.1) is also easy and can be found in [8, page 12].

1.4 Research Leading to PCP Theorem

The roots of PCP theorem go back to the Interactive Proofs introduced by Goldwasser, Micali, Rackoff [53] and Arthur-Merlin games introduced by Babai [14]. Both these proof systems feature a polynomial time probabilistic verifier interacting with an all-powerful adversary called prover. Techniques from program checking due to Blum, Kannan [19], Lipton [89] and Blum, Luby, Rubinfeld [20], as well as ideas about representing logical formulae with polynomials (Lund, Fortnow, Karloff, Nisan [92] and Shamir [108]) were used to show surprising results that IP = PSPACE ([92], [108]) and MIP = NEXPTIME (Babai, Fortnow, Lund [15]). The result MIP = NEXPTIME was *scaled down* to $\mathrm{NP} \subseteq \mathrm{PCP}[O(\log n \log\log n), O(\log n \log\log n)]$ by Feige, Goldwasser, Lovász, Safra and Szegedy [38]. They also observed the connection between PCPs and inapproximability, showing a strong inapproximability result for Clique. The PCP theorem was proved in a sequence of two papers by Arora, Safra [13] and Arora, Lund, Motwani, Sudan, Szegedy [12].

1.5 Progress after PCP Theorem

PCP Theorem has given a tremendous boost to the theory of inapproximability. Hardness results (in many cases optimal) are now known for a variety of optimization problems. In this section, we point out some high-level techniques used to prove such results. The list of results stated here is by no means comprehensive, we give only a few examples to illustrate each technique.

Theorem 1.3.1 gives a "gap" instance of MAX-3SAT (let us call it Gap-3SAT). A general method to prove hardness of some other problem A is to give a "gap-preserving" reduction from Gap-3SAT to the problem A. Let us assume A is a minimization problem. A gap-preserving reduction is a poly-time reduction that maps an instance ψ of Gap-3SAT to an instance I of problem A with the following property :

- (Completeness :) If ψ is satisfiable, then $OPT(I) \leq d$
- (Soundness :) If no more than a fraction s of clauses of ψ are satisfiable, then $OPT(I) \geq Cd$.

Such a reduction implies that it is NP-hard to approximate A within factor C. [1]

As a bottomline, all hardness results are essentially gap-preserving reductions from Gap-3SAT. These reductions are often complicated, involving a sequence of reductions. The first step is to "amplify the gap" of the Gap-3SAT instance using Raz's Parallel Repetition Theorem [103]. This gives a 2-Prover-1-Round game which can be equivalently viewed either as (i) the Label Cover problem defined by Arora et al [9] (see Definition 2.3.1) or as (ii) PCPs where the verifier reads two symbols from the proof and the symbols come from a (big) alphabet of constant size.

Label Cover problem is a cleanly defined combinatorial problem and abstracts out the essential properties of 2-Prover games. Therefore we find it convenient to work with this problem in this thesis and use it as a canonical problem to reduce from for most of

[1] Many variants of gap-preserving reduction are defined, e.g. L-reductions [101] in the definition of class MAX-SNP and E-reductions [73] used to relate classes APX and MAX-SNP. Here we give a somewhat non-rigorous definition that suffices for most purposes.

the hardness results. We classify techniques for showing hardness results into four broad categories :

(1) Direct Reduction from Label Cover

Examples in this category include

- $(1-\epsilon)\ln n$ hardness for Set Cover by Feige [37], building on the work of Lund and Yannakakis [93].
- Hardness of Closest Vector Problem and Closest Codeword Problem by Arora, Babai, Stern and Sweedyk [9].
- Hardness of Shortest Vector Problem in this thesis (Chapter 6 and [78]).
- $\Omega(\log^{2-\epsilon} n)$ hardness for Group Steiner Tree by Halperin and Krauthgamer [57]. This introduces a new technique of building a gadget from an integrality gap example for a natural linear program.

(2) Using Long Codes and Fourier Analysis

Many results, most notably results on constraint satisfaction problems (CSPs), fall into this category. These results are proved using a framework developed in the proof of PCP Theorem and then by Bellare, Goldreich, Sudan [16] and Håstad ([59], [60]). The idea is that the verifier expects as a proof, encodings of labels for the Label Cover problem and then runs some tests to check consistency between these encodings. An encoding scheme called *Long Code* was introduced in [16] and Håstad introduced the powerful technique of analyzing tests with Fourier analysis. A detailed presentation of this methodology appears in Chapter 2. Examples include

- Håstad's $n^{1-\epsilon}$ hardness for Clique [59], $\frac{8}{7}-\epsilon$ hardness for MAX-3SAT, and $2-\epsilon$ hardness for Max-3-Lin-2 [60].

- Hardness of boolean k-CSPs by Samorodnitsky and Trevisan [106].

- Hardness of coloring 4-uniform hypergraphs by Guruswami, Håstad and Sudan [55].

- Hardness of coloring 3-uniform and 4-uniform hypergraphs in this thesis (Chapters 4, 5 and [75], [76] respectively).

- Hardness of Vertex Cover on 4-uniform hypergraphs by Holmerin [66].

(3) Using Combinatorial View of Long Code

This technique was introduced in Dinur and Safra's paper [32]. The idea is to take a combinatorial view of the Long Code and use instead of Fourier analysis, theorems from extremal combinatorics and sensitivity analysis of boolean functions. Sensitivity analysis itself relies on Fourier analysis, so there is no strict distinction between this technique and technique in Category (2). Examples include,

- 1.36-hardness result for Vertex Cover by Dinur and Safra [32].

- $k-1-\epsilon$ hardness for Vertex Cover on k-uniform hypergraphs in this thesis (Chapter 7 and [29]).

- Hardness of coloring 3-uniform hypergraphs by Dinur, Regev and Smyth [31].

(4) Reductions from Problems in Previous Categories

Here we include results obtained by a reduction from problems whose hardness is proved using the above three techniques. One typically uses clever gap-preserving gadgets. The techniques are varied, problem-dependent and often quite involved. It is beyond the scope of this thesis to give a survey of these techniques. Examples include,

- $\frac{17}{16} - \epsilon$ hardness for MAX-CUT by Håstad [60], via a reduction from Max-3-Lin-2.
- $\frac{117}{116} - \epsilon$ hardness for Asymmetric TSP by Papadimitriou and Vempala [100], via a reduction from Max-3-Lin-2.
- $\Omega(\log^* n)$-hardness for Asymmetric k-Center by Chuzhoy et al [26], via a reduction from Hypergraph Vertex Cover.
- $\sqrt{2} - \epsilon$ hardness for Shortest Vector Problem in L_2 norm by Micciancio [96], via a reduction from Closest Vector Problem.

Chapter 2

A Framework for Building PCPs

Techniques in this thesis can be best viewed in context of a general framework for designing PCPs with only a few query bits. In general form, the framework goes back to the proof of PCP Theorem by Arora, Safra [13] and Arora et al [12]. Some of the key ideas in their work are composition of verifiers and different encoding schemes. Bellare, Goldreich and Sudan [16] invented the Long Code which was then used to a great effect by Håstad ([59], [60]). In this chapter, we explain this framework by presenting a complete proof of Håstad's 3-Bit PCP. In the rest of the thesis, we extend the framework in many ways, using new verifiers and codes at various levels.

We present the following result in this chapter.

Theorem 2.0.1 *(Håstad's 3-Bit PCP [60])*

$$\forall\, \epsilon, \delta > 0, \qquad \mathrm{NP} = \mathrm{PCP}_{1-\epsilon, \frac{1}{2}+\delta}[O(\log n), 3]$$

In other words, NP *has a probabilistic poly-time verifier that uses logarithmic randomness, reads* 3 *bits from the proof, has completeness* $\geq 1 - \epsilon$ *and soundness* $\leq \frac{1}{2} + \delta$ *where*

ϵ, δ can be made arbitrarily small. Moreover, the verifier accepts if and only if the 3 bits read from the proof satisfy a linear predicate.

Let us say we have proved this theorem. Now let the bits in the proof correspond to (unknown) boolean variables so that a proof corresponds to a boolean assignment to these variables. For a fixed choice of the random coins, the verifier reads 3 bits from the proof, say x, y, z and accepts if and only if $x \oplus y \oplus z = 0$ (or $x \oplus y \oplus z = 1$). Write down poly(n) such equations for all possible choices of $O(\log n)$ random coins. The correspondence between proofs and boolean assignments shows that the probability of accepting a proof equals the fraction of equations satisfied by an assignment. Using completeness and soundness properties given by Theorem 2.0.1, we get :

Theorem 2.0.2 *Let Max-3-Lin-2 be the following problem : Given a system of linear equations mod 2, each equation containing 3 variables, find an assignment that satisfies maximum number of equations. Then for every $\epsilon, \delta > 0$, there is a poly-time reduction from any language $L \in \mathrm{NP}$ to Max-3-Lin-2 such that :*

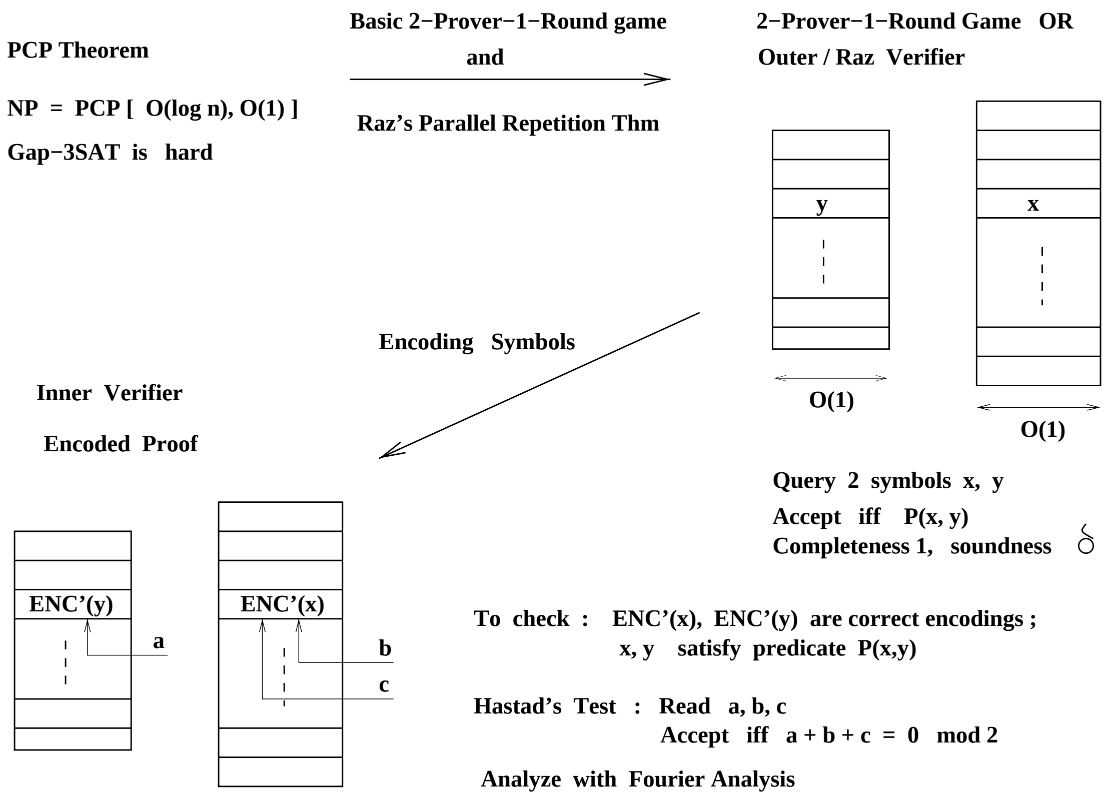
- *The reduction maps input x to an instance Γ of Max-3-Lin-2.*
- *If $x \in L$, then Γ has an assignment that satisfies $1 - \epsilon$ fraction of equations.*
- *If $x \notin L$, then no assignment to Γ satisfies more than a fraction $\frac{1}{2} + \delta$ of equations.*

In particular, it is NP-hard to approximate Max-3-Lin-2 within factor $2 - \epsilon$. This result is tight since a random assignment to variables satisfies half the equations.

It is now clear how constructing a specific PCP implies a hardness result for a constraint satisfaction problem. The constraints (in this case linear) correspond to the acceptance predicate of the verifier and the hardness factor equals the ratio between completeness and soundness parameters.

2.1 Plan for Building 3-Bit PCP

The overall plan for proving Theorem 2.0.1 is as follows (see Fig 2.1) : We start with the PCP Theorem (Theorem 1.3.1) which shows hardness of Gap-3SAT. A simple 2-Prover-1-Round game is constructed from Gap-3SAT and then the gap is amplified using Raz's Parallel Repetition Theorem. This results in a 2-Prover-1-Round game where the provers' answers are of constant size, the game has perfect completeness and very low soundness. The answers are from an alphabet of size $1/\delta^{O(1)}$ where δ is the soundness of the 2-Prover game.



PCP Theorem
NP = PCP [O(log n), O(1)]
Gap−3SAT is hard
Basic 2−Prover−1−Round game and
Raz's Parallel Repetition Thm
2−Prover−1−Round Game OR
Outer / Raz Verifier
y
x
O(1)
O(1)
Query 2 symbols x, y
Accept iff P(x, y)
Completeness 1, soundness δ
Encoding Symbols
Inner Verifier
Encoded Proof
ENC'(y)
ENC'(x)
a
b
c
To check : ENC'(x), ENC'(y) are correct encodings ;
x, y satisfy predicate P(x,y)
Hastad's Test : Read a, b, c
Accept iff a + b + c = 0 mod 2
Analyze with Fourier Analysis

Figure 2.1: Overview of Håstad's 3-Bit PCP

The 2-Prover game can be easily converted to a PCP where the verifier reads only 2 symbols from the proof, has perfect completeness and soundness δ. The verifier demands provers' answers to all possible questions as a written proof. Instead of asking questions to provers and receiving answers, the verifier reads off these answers from the written proof. The completeness/soundness properties follow from the properties of the 2-Prover game. This verifier is usually referred to as the "Outer Verifier" or "Raz Verifier".

Now the trouble is that the Outer Verifier reads symbols from a huge alphabet (of size $1/\delta^{O(1)}$) and our goal is to construct a verifier that reads just 3 *bits*. This is achieved by expecting as a proof, encodings of provers' answers instead of the answers themselves. We fix an encoding scheme ENC (which is most often the Long Code). The new verifier, usually referred to as the "Inner Verifier", asks for the encodings $ENC(x)$ for answers x of the provers. The Outer Verifier would read two answers x, y and accept if a certain predicate $P(x, y)$ is satisfied. The Inner Verifier however has access to strings $ENC'(x), ENC'(y)$ which he expects to be the encodings $ENC(x), ENC(y)$ respectively. A crucial point is that a cheating proof could contain strings $ENC'(x), ENC'(y)$ which might be arbitrary strings and may have nothing to do with the correct encodings. It is verifier's task to guard against such cheating proofs. The verifier needs to verify the following in a probabilistic sense :

- $ENC'(x), ENC'(y)$ are "close" to the true encodings $ENC(x), ENC(y)$ respectively. This is referred to as the *Codeword Test*.

- $P(x, y)$ is satisfied. This is referred to as the *Consistency Test*.

This task looks hopeless given the restriction that the verifier is allowed to read only 3 bits from $ENC'(x), ENC'(y)$. The beauty of Håstad's construction lies in combining these tasks in one single 3-bit test and analyze the test with Fourier Analysis. It can be shown that if the test accepts with probability little more than $\frac{1}{2}$, then there is a way to "decode" the strings $ENC'(x), ENC'(y)$ and define provers' answers which make the verifier in the 2P1R protocol accept with a decent probability. However, this would be a contradiction, provided the 2P1R game was chosen to have very low soundness.

In the following sections, we carry out this high level plan to prove Theorem 2.0.1. We define a problem called the Label Cover problem which is an equivalent way of looking at 2-Prover games. The PCPs in this thesis are described in terms of the Label Cover problem; we find it easier for the sake of presentation. Let us restate the PCP Theorem :

Theorem 2.1.1 *There exists an absolute constance $c < 1$ such that it is NP-hard to distinguish satisfiable 3SAT formulae (YES instances) from those where only a fraction c of the clauses can be satisfied by any assignment (NO instances). This formula has a regular structure, i.e. any clause is of length exactly 3 and any variable appears in exactly 5 clauses. We call the instances of 3SAT given by this theorem as instances of Gap-3SAT-5.*

Remark : Compared to Theorem 1.3.1, we have an extra requirement that the 3SAT formula has a regular structure. This is easy to achieve, see e.g. [37]. This property is usually convenient, but not essential.

2.2 The 2-Prover 1-Round Game

Definition 2.2.1 *A 2P1R game $\mathcal{L}_{2p1r}(V, W, N, M, D, Q)$ consists of a probabilistic verifier V_{2p1r} and two provers P_1 and P_2.*

- *V, W are sets of questions the verifier can ask the two provers respectively.*
- *N, M are sets of provers' answers. A* strategy *of the provers is a map $\Phi : V \mapsto N, \Phi : W \mapsto M$.*

- $Q : V \times W \times N \times M \mapsto \{0, 1\}$ *is the acceptance predicate of the verifier.*
- D *is a probability distribution on* $V \times W$.

The verifier picks a pair of questions $(v, w) \in V \times W$ *with distribution* D *and sends questions* v, w *to provers* P_1, P_2 *respectively. The provers return answers* $\Phi(v), \Phi(w)$ *according to their strategy. The verifier accepts if and only if* $Q(v, w, \Phi(v), \Phi(w)) = 1$. *The value of the game is defined to be the maximum probability with which the provers can make the verifier accept over all possible prover strategies.*

We build a basic 2P1R game from a Gap-3SAT-5 formula ϕ with variables $\{x_1, x_2, \dots, x_n\}$ and clauses $\{C_1, C_2, \dots, C_m\}$.

Basic 2-Prover Game

1. The verifier chooses a clause C_k uniformly at random and a variable x_j uniformly at random, appearing in C_k. The verifier sends x_j to prover P_1 and C_k to prover P_2.
2. The verifier receives a value for x_j from P_1 and a satisfying assignment for the clause C_k from P_2. The verifier accepts if and only if the value for x_j agrees with the assignment to C_k.

Note that the strategy of P_1 is just an assignment to variables of ϕ. Let us say this assignment satisfies a fraction c' of the clauses. Then for every clause C satisfied by the assignment, P_2's answer could be consistent with the assignment. However, if a clause C is unsatisfied, then P_2 has to flip the value of at least one variable in C and then the verifier will catch this "cheating" with probability $1/3$. Thus the optimal strategy convinces the verifier with probability $c' + (1 - c')\frac{2}{3} = (2 + c')/3$. It follows that if ϕ is a YES instance of Gap-3SAT-5, the provers have a strategy with value 1 (completeness) and otherwise the optimal strategy of provers has value at most $(2+c)/3 < 1$ (soundness) where c is the constant from Theorem 2.1.1.

Parallel Repetition and Raz Verifier

One can reduce the soundness of this game by running the verifier u times in parallel. We call the new verifier Raz Verifier. In this new game, the verifier picks a set of u clauses at random, say $w = \{C_i \mid i = 1, 2, \ldots, u\}$, and asks P_2 to give a satisfying assignment to these clauses. For every i, the verifier picks a variable x_i at random from the clause C_i. Let $v = \{x_i | i = 1, 2, \ldots, u\}$ be the set of these variables. The verifier asks P_1 to give an assignment to the set v. The verifier accepts if and only if the answers of the two provers agree on the set of variables v. If we denote $\pi^{\,v,w}$ to be the projection that maps an assignment to w to its sub-assignment to v, the verifier accepts if and only if the answer of P_2 is σ and the answer of P_1 is $\pi^{\,v,w}(\sigma)$.

Clearly, if the Gap-3SAT-5 formula is a YES instance, then provers have a strategy to make the verifier accept with probability 1 (completeness). The Parallel Repetition Theorem of Raz [103] gives the soundness property. It shows that if Gap-3SAT-5 instance is a NO instance, no strategy can make the verifier accept with probability more than d_c^u for some absolute constant $d_c < 1$. Let us formulate this result for future reference.

Theorem 2.2.2 *If only a fraction $c < 1$ of the clauses of ϕ can be simultaneously satisfied, then no strategy of P_1 and P_2 can make the verifier accept with probability greater than d_c^u. Here $d_c < 1$ is a constant that only depends on c.*

2.3 The Label Cover Problem

For the sake of presentation, it is easier to work with a problem called Label Cover problem instead of 2P1R games. We follow the Label Cover terminology throughout this thesis.

Definition 2.3.1 *A Label Cover problem $\mathcal{L}(G(V,W,E),N,M,\{\pi^{v,w}|(v,w)\in E\})$ consists of a regular bipartite graph $G(V,W,E)$ with bipartition V,W. Every vertex in V is supposed to get a label from a set N and every vertex in W is supposed to get a label from a set M. With every edge $(v,w)\in E$, there is associated a "projection" $\pi^{v,w}: M \rightarrow N$. For an assignment of labels to the vertices of the graph, that is for a function $\Phi : V \mapsto N$, $\Phi : W \mapsto M$, an edge (v,w) is said to be satisfied if $\pi^{v,w}(\Phi(w)) = \Phi(v)$. The goal is to find an assignment of labels that maximizes the number of satisfied edges. We define $OPT(\mathcal{L})$ to be the maximum fraction of edges satisfied by any labeling.*

It is clear that the Label Cover problem is the same as a 2P1R game (see Definition 2.2.1). Take V,W as the sets of questions to be asked to the provers, N,M as the sets of possible answers, $(v,w)\in E$ as the pairs of questions to be asked to the provers, and $\pi^{v,w}$ as the acceptance predicate of the verifier. Using this correspondence, Theorem 2.2.2 can be restated as :

Theorem 2.3.2 *There is an absolute constant $\gamma > 0$ such that for every integer parameter u, it is NP-hard to distinguish between the following two cases : A Label Cover problem $\mathcal{L}(G(V,W,E),N,M,\{\pi^{v,w}\})$ with $|M| = 7^u$, $|N| = 2^u$ has*

- $OPT(\mathcal{L}) = 1$ *OR*
- $OPT(\mathcal{L}) \leq 2^{-\gamma u}$

It can be assumed that $G(V, W)$ is a regular bipartite graph where every vertex in V has degree 5^u and every vertex in W has degree 3^u.

2.4 Long Codes and Fourier Analysis

The long code was introduced by Bellare et al [16] and the use of Fourier analysis in PCP setting was initiated by Håstad. In this section, we gives the basics of long codes and their Fourier analysis. As will be clear later, it is convenient to work with bits valued ± 1 instead of $\{0, 1\}$ with the correspondence $0 \mapsto 1$, $1 \mapsto -1$.

Definition 2.4.1 *The long code over a finite domain M is indexed by all functions $g : M \mapsto \{-1, 1\}$. The long code B of an element $y \in M$ is given by*

$$B(g) := g(y) \quad \forall \; g : M \mapsto \{-1, 1\}$$

Let $\mathcal{G} := \{g \mid g : M \mapsto \{-1, 1\}\}$. Consider the space of all "tables" $B : \mathcal{G} \mapsto \mathbb{R}$. In particular, a long code is one such table. Consider the characters χ_β where $\beta \subseteq M$. There is one such character for every β. The character χ_β is a table defined by

$$\chi_\beta(g) := \prod_{y \in \beta} g(y)$$

The following identities are easily checked,

Lemma 2.4.2 *For $\beta, \gamma \subseteq M$ and $g, h : M \mapsto \{-1, 1\}$,*

- $\chi_\beta(gh) = \chi_\beta(g)\chi_\beta(h)$
- $\chi_\beta(g)\chi_\gamma(g) = \chi_{\beta\Delta\gamma}(g)$ *where Δ denotes the symmetric difference of sets.*

- $E_g[\chi_\beta(g)] = 1$ *if* $\beta = \emptyset$ *and* 0 *otherwise.*

For tables B_1, B_2, define their inner product as

$$< B_1, B_2 > := \frac{1}{2^{|M|}} \sum_{g \in \mathcal{G}} B_1(g) B_2(g) = E_g\big[B_1(g)B_2(g)\big]$$

Let us check that the characters form an orthonormal basis under this definition of inner product. The number of characters is $2^{|M|}$ which equals the dimension of the space of tables. They are orthonormal because,

$$\begin{aligned} < \chi_\beta, \chi_\gamma > &= E_g[\chi_\beta(g)\chi_\gamma(g)] \\ &= E_g[\chi_{\beta\Delta\gamma}(g)] \\ &= 1 \text{ if } \beta = \gamma \qquad \text{and } 0 \text{ if } \beta \neq \gamma \end{aligned}$$

It follows that any table can be expanded as $B = \sum_{\beta \subseteq M} \widehat{B}_\beta \chi_\beta$ where $\widehat{B}_\beta$ are the Fourier coefficients with $\sum_\beta |\widehat{B}_\beta|^2 = < B, B >$ (Parseval's identity). When $B : \mathcal{G} \mapsto \{-1, 1\}$, we have $\sum_\beta |\widehat{B}_\beta|^2 = 1$. The Fourier coefficients are given by

$$\widehat{B}_\beta = < B, \chi_\beta > = E_g\big[B(g)\chi_\beta(g)\big]$$

When $B : \mathcal{G} \mapsto \{-1, 1\}$, it is clear that B agrees with the character function χ_β on $\frac{1+\widehat{B}_\beta}{2}$ fraction of inputs. Thus a Fourier coefficient is a measure of how close B is to a character function. When $\beta = \emptyset$, the value of the coefficient $\widehat{B}_\emptyset$ is just $E_g[B(g)]$.

The following lemma introduces a notion called "folding" which turns out to be crucial in the analysis.

Definition 2.4.3 *A table B is folded if $B(-g) = -B(g)$ for any g.*

Lemma 2.4.4 *If B is folded and $\widehat{B}_\beta \neq 0$ then $|\beta|$ is odd and in particular β is a non-empty set.*

Proof:

$$\begin{aligned} \widehat{B}_\beta &= E_g[B(g)\chi_\beta(g)] = E_g[B(-g)\chi_\beta(-g)] \\ &= E_g[(-B(g))\prod_{y\in\beta}(-g(y))] = -(-1)^{|\beta|}E_g[B(g)\chi_\beta(g)] = -(-1)^{|\beta|}\widehat{B}_\beta \end{aligned}$$

Thus $\widehat{B}_\beta = 0$ unless $|\beta|$ is odd. ■

To make sure that an arbitrary table is folded we access the table as follows. For each pair $(g, -g)$ we choose (in some arbitrary but fixed way) one representative. If g is chosen, then the value of the table required at g is accessed as usual by reading $B(g)$. If the value at $-g$ is required then $B(g)$ is read and the result is negated. If $-g$ is chosen from the pair, the procedures are reversed.

We need one more lemma that relates boolean functions on domain M to functions on domain N via a projection map $\pi : M \mapsto N$. Let $f : N \mapsto \{-1, 1\}$ and $(f \circ \pi) : M \mapsto \{-1, 1\}$ be defined as

$$(f \circ \pi)(y) = f(\pi(y)) \quad \forall\ y \in M$$

For a set $\beta \subseteq M$, let $\pi(\beta) \subseteq N$ denote the projected set

$$\pi(\beta) := \{\pi(y) \mid y \in M\}$$

Also define a set $\pi_2(\beta) \subseteq \pi(\beta)$ as the set of elements of N that have an odd number of pre-images in β. To be precise,

$$\pi_2(\beta) := \{x \in N \mid |\{y \in \beta \mid \pi(y) = x\}| \text{ is } \text{ odd } \}$$

Lemma 2.4.5 $\quad \chi_\beta(f \circ \pi) = \chi_{\pi_2(\beta)}(f)$

2.5 Håstad's 3-Bit PCP

In this section, we describe the 3-Bit PCP. Overall plan for the construction has already been outlined in Section 2.1. For some intuition behind our construction and a general approach for PCP design, see Sections 2.5.1 and 2.6.

The verifier starts with an instance $\mathcal{L}(G(V, W, E), N, M, \{\pi^{v,w}\})$ of the Label Cover problem given by Theorem 2.3.2. He expects as a proof, the long code of the label of every $v \in V$ and $w \in W$ in a supposedly correct labeling to $\mathcal{L}$. These long codes are assumed to be folded. The verifier picks an edge (v, w) of Label Cover at random. Let A, B be supposed long codes of labels of v, w respectively. The verifier has to check that A, B are indeed correct long codes and they encode labels $a \in N, b \in M$ respectively, with $\pi^{v,w}(b) = a$. The verifier checks the proof as follows (see Fig. 2.2) :

1. Pick a random $w \in W$ and its random neighbor $v \in V$. Let $\pi = \pi^{v,w} : M \mapsto N$ be the corresponding projection map.

2. Let A, B be the supposed long codes of labels of v and w resp. Recall that B is indexed by all functions $g : M \mapsto \{-1, 1\}$ and A is indexed by all functions $f : N \mapsto \{-1, 1\}$.

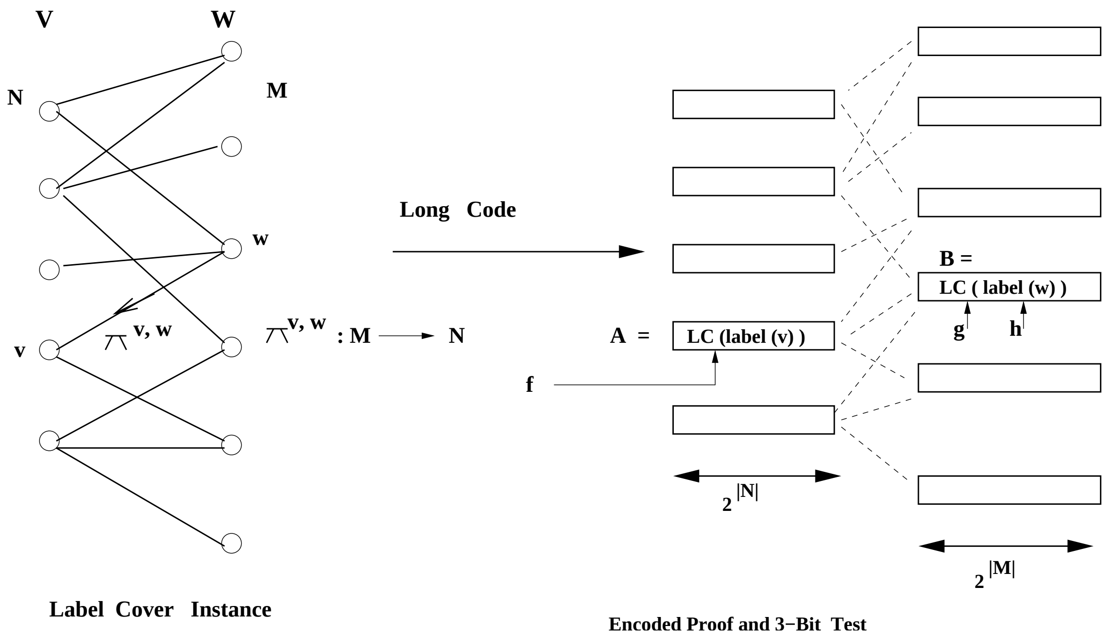


Figure 2.2: Label Cover, Long Code and 3-Bit PCP

3. Pick a random function $g : M \mapsto \{-1, 1\}$ and a random function $f : N \mapsto \{-1, 1\}$.

4. Pick a function $\mu : M \mapsto \{-1, 1\}$ by defining independently for every $y \in M$,

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

Thus $\mu()$ is a function that is -1 on about ϵ fraction of inputs chosen randomly.

5. Let $h : M \mapsto \{-1, 1\}$ be a function where for every $y \in M$, we define

$$h(y) := g(y) \cdot f(\pi(y)) \cdot \mu(y)$$

In other words, $h = g(f \circ \pi)\mu$.

6. Read bits $A(f), B(g), B(h)$ from the proof. Accept if and only if

$$A(f)B(g)B(h) = 1$$

Remark : In $\{0,1\}$ notation, the verifier's test is $A(f) \oplus B(g) \oplus B(h) = 0$ which is indeed a linear test.

2.5.1 Completeness

We show that the completeness of this test is $1-\epsilon$. In a correct labeling to $\mathcal{L}$, let a, b be labels of v, w respectively, with $\pi(b) = a$. In a correctly encoded proof, A, B are long codes of a, b respectively and therefore $A(f) = f(a), B(g) = g(b)$, and $B(h) = h(b) = g(b)f(\pi(b))\mu(b) = g(b)f(a)\mu(b)$. Thus the test accepts provided $\mu(b) = 1$ which happens with probability $1-\epsilon$.

Let us see the intuition behind such a test. We want to test consistency between tables A and B. The test reads one bit from table A and two bits from table B, say $A(f), B(g), B(h)$. The test is required to be linear, so the acceptance predicate should be $A(f)B(g)B(h) = 1$. Designing a test means specifying a distribution with which the triple (f, g, h) is picked. Completeness condition requires that if B is long code of *any* $y \in M$ and A is long code of $\pi(y)$, then the test should accept with probability 1 (or close to 1). In this case, $A(f) = f(\pi(y)), B(g) = g(y), B(h) = h(y)$ and we require that $f(\pi(y))g(y)h(y) = 1$. In other words, $h = g(f \circ \pi)$.

The reason for introducing error function $\mu()$ is quite subtle. Let us say $h = g(f \circ \pi)$. We will show that the test would accept with probability 1 even if B is not a long code. Fix $x_0 \in N$ and let A be long code of x_0. Let $\beta \subseteq M$ be any set with odd cardinality such that all elements of β map to x_0 under projection π. Let $B = \chi_\beta$. Note that if $|\beta| = 1$

then B would be a long code. So when $|\beta|$ is "large", B is interpreted as being "far" from any long code. The test accepts because

$$A(f)B(g)B(h) = f(x_0)\chi_\beta(gh) = f(x_0)\prod_{y\in\beta} g(y)h(y) = f(x_0)\prod_{y\in\beta} g(y)g(y)f(\pi(y)) = 1$$

since $|\beta|$ is odd. Thus the test with $h = g(f \circ \pi)$ fails. We need to somehow enforce the condition that β is singleton, or at least of "small" size. The error function $\mu()$ achieves exactly this.

So we let $h = g(f \circ \pi)\mu$. We will see that if the test accepts with probability $\frac{1}{2} + \delta$, then Fourier coefficients of tables A and B satisfy :

$$\sum_{\alpha,\beta:\alpha=\pi_2(\beta)} \widehat{A}_\alpha \widehat{B}_\beta^2 \,(1-2\epsilon)^{|\beta|} \geq \delta$$

The condition $\alpha = \pi_2(\beta)$ captures consistency between A, B and this is precisely the role of $f \circ \pi$ in the test. Role of $\mu()$ is to introduce the factor $(1-2\epsilon)^{|\beta|}$. Thus terms with $|\beta| \geq O(\frac{1}{\epsilon}\log(1/\delta))$ contribute only negligible amount, so one can as well assume that the sum is restricted to β with smaller size. The idea of introducing $\mu()$ is almost a magic trick. One loses perfect completeness, but it enables us to bound $|\beta|$ in the soundness analysis.

2.5.2 Soundness

Now we show that if the test accepts with probability $\frac{1}{2}+\delta$, then there exists an assignment of labels to $\mathcal{L}$ that satisfies a fraction $4\epsilon\delta^2$ of the edges. Thus if we choose the parameter u of Theorem 2.3.2 large enough so that $2^{-\gamma u} < 4\epsilon\delta^2$, it follows that the soundness of the verifier is at most $\frac{1}{2} + \delta$.

The idea is to write the probability of acceptance as the following arithmetic expression :

$$\Pr[\text{accept}] = E_{v,w,f,g,\mu}\big[\frac{1 + A(f)B(g)B(h)}{2}\big] \tag{2.1}$$

Note that when the test accepts, the expression inside the bracket equals 1 and equals 0 otherwise. Therefore, Pr[accept] is the expectation of this expression over the random choices made by the verifier. It follows that if $\Pr[\text{accept}] \geq \frac{1}{2} + \delta$, then

$$E_{v,w,f,g,\mu}\big[A(f)B(g)B(h)\big] \geq \delta$$

We write the tables A and B as their Fourier expansions, i.e.

$$A(f) = \sum_{\alpha \subseteq N} \widehat{A}_\alpha \chi_\alpha(f) \qquad B(g) = \sum_{\beta \subseteq M} \widehat{B}_\beta \chi_\beta(g) \qquad B(h) = \sum_{\gamma \subseteq M} \widehat{B}_\gamma \chi_\gamma(h)$$

where $\widehat{A}_\alpha, \widehat{B}_\beta, \widehat{B}_\gamma$ are Fourier coefficients. Substituting,

$$E_{v,w,f,g,\mu}\big[\sum_{\alpha,\beta,\gamma} \widehat{A}_\alpha \widehat{B}_\beta \widehat{B}_\gamma \chi_\alpha(f)\chi_\beta(g)\chi_\gamma(h)\big] \geq \delta$$

Using Lemmas 2.4.2 and 2.4.5,

$$\chi_\gamma(h) = \chi_\gamma(g(f \circ \pi)\mu) = \chi_\gamma(g)\chi_\gamma(f \circ \pi)\chi_\gamma(\mu) = \chi_\gamma(g)\chi_{\pi_2(\gamma)}(f)\chi_\gamma(\mu)$$

Thus we get

$$E_{v,w,f,g,\mu}\big[\sum_{\alpha,\beta,\gamma} \widehat{A}_\alpha \widehat{B}_\beta \widehat{B}_\gamma \chi_\alpha(f)\chi_\beta(g)\chi_\gamma(g)\chi_{\pi_2(\gamma)}(f)\chi_\gamma(\mu)\big] \geq \delta$$

which can be simplified to

$$E_{v,w,f,g,\mu}\big[\sum_{\alpha,\beta,\gamma}\widehat{A}_\alpha\widehat{B}_\beta\widehat{B}_\gamma\chi_{\alpha\Delta\pi_2(\gamma)}(f)\chi_{\beta\Delta\gamma}(g)\chi_\gamma(\mu)\big]\geq\delta \tag{2.2}$$

By Lemma 2.4.2, the expectation $E_g[\chi_{\beta\Delta\gamma}(g)]$ vanishes unless $\beta\Delta\gamma = \emptyset$, i.e. unless $\beta=\gamma$. Similarly, the expectation over f vanishes unless $\alpha=\pi_2(\gamma)$. Also $E_\mu[\chi_\gamma(\mu)] = (1-2\epsilon)^{|\gamma|}$ by the following lemma.

Lemma 2.5.1 $E_\mu[\chi_\gamma(\mu)] = (1-2\epsilon)^{|\gamma|}$

Proof:

$$E_\mu[\chi_\gamma(\mu)] = E_\mu[\prod_{y\in\gamma}\mu(y)] = \prod_{y\in\gamma}E_\mu[\mu(y)] = (1-2\epsilon)^{|\gamma|}$$

since, for every y, we have $\mu(y)=1$ with probability $1-\epsilon$ and $\mu(y)=-1$ with probability ϵ. ■

Thus Equation (2.2) can be written as

$$E_{v,w}\big[\sum_\beta\widehat{A}_{\pi_2(\beta)}\widehat{B}^2_\beta(1-2\epsilon)^{|\beta|}\big]\geq\delta \tag{2.3}$$

Using Cauchy-Schwartz inequality and the fact that $(1-2\epsilon)^{2|\beta|}\leq(e^{-2\epsilon})^{2|\beta|}=e^{-4\epsilon|\beta|}\leq\frac{1}{4\epsilon|\beta|}$,

$$\begin{aligned}
\delta &\leq E_{v,w}\left[\left|\sum_\beta\widehat{A}_{\pi_2(\beta)}\widehat{B}_\beta(1-2\epsilon)^{|\beta|}\ \cdot\ \widehat{B}_\beta\right|\right]\\
&\leq E_{v,w}\left[\sqrt{\sum_\beta\widehat{A}^2_{\pi_2(\beta)}\widehat{B}^2_\beta(1-2\epsilon)^{2|\beta|}}\sqrt{\sum_\beta\widehat{B}^2_\beta}\right]\\
&\leq E_{v,w}\left[\sqrt{\sum_\beta\widehat{A}^2_{\pi_2(\beta)}\widehat{B}^2_\beta(4\epsilon|\beta|)^{-1}}\right]\quad\text{since }\sum_\beta\widehat{B}^2_\beta=1
\end{aligned}$$

Again applying Cauchy-Schwartz, we get

$$E_{v,w}\left[\sum_{\beta}\widehat{A}^2_{\pi_2(\beta)}\widehat{B}^2_{\beta}\frac{1}{|\beta|}\right] \geq 4\epsilon\delta^2 \tag{2.4}$$

Now consider the following (randomized) way of assigning labels to vertices of $\mathcal{L}$. For every $w \in W$, let B be the supposed long code for w in the proof. Pick the Fourier coefficient β with probability $\widehat{B}^2_{\beta}$, pick an element $y \in \beta$ at random and define label$(w) = y$. Similarly, for every $v \in V$, let A be the supposed long code for v in the proof. Pick the Fourier coefficient α with probability $\widehat{A}^2_{\alpha}$, pick an element $x \in \alpha$ at random and define label$(v) = x$. Inequality (2.4) says that there is a good chance that we will have $\alpha = \pi_2(\beta)$ and after picking $x \in \alpha$, with chance $1/|\beta|$ pick $y \in \beta$ such that $\pi(y) = x$. Thus the expected fraction of edges of the Label Cover instance satisfied by this (randomized) labeling is at least $4\epsilon\delta^2$. Hence the Label Cover instance has a labeling that satisfies this much fraction of edges, completing the proof.

2.6 A General Approach to PCP Design

We use the proof of Håstad's 3-Bit PCP to illustrate some general principles for designing PCPs.

The acceptance predicate for a PCP test is tailor-made for the target problem for which we desire a hardness of approximation result. For example, to show hardness of Max-3-Lin-2, we designed a test with predicate $x \oplus y \oplus z = 0$. This is a predicate with arity three and a binary alphabet. In general, the arity and alphabet size could be arbitrary. The following table gives acceptance predicates for some well-known problems.

Problem	**Predicate**	**Alphabet Size**	**Folding**
MAX-2SAT	$x \vee y$	2	YES
MAX-3SAT	$x \vee y \vee z$	2	YES
MAX-CUT	$x \neq y$	2	NO
MAX-k-CUT	$x \neq y$	k	NO
3-Colorability of graphs	$x \neq y$	3	NO
2-Colorability of 4-uniform hypergraphs	Not-All-Equal(x, y, z, w)	2	NO
Vertex cover in graphs	$x \vee y$	2	NO

Acceptance predicates for NP-hard problems

Thus the choice of acceptance predicate is straightforward. The most intricate part is designing the actual test. One needs to check consistency between two long codes A and B. More specifically, one needs to check that A and B encode labels a and b respectively such that $\pi(b) = a$ for some given map π. The test corresponds to selecting bits from tables A and B in an appropriate way. Choice of these bits is dictated by the completeness requirement, namely that a correct proof be accepted with probability 1 (or close to 1). In the 3-Bit PCP, we designed a verifier that reads three bits, $A(f), B(g), B(h)$. As seen in Section 2.5.1, the completeness requirement suggests that h be defined as $g(f \circ \pi)$. Then a counter-example is presented to motivate the use of error function $\mu()$ resulting in imperfect completeness. Some applications however require perfect completeness, e.g. hypergraph coloring results require that in completeness case, *every* edge is non-monochromatic. In such applications, our test must accept with probability 1.

For the soundness analysis, we begin by arithmetizing the acceptance probability of the verifier as in Equation (2.1). We plug in Fourier expansions, do straightforward simplifications and get Equation (2.3). We use such an equation to extract some meaningful consistency between tables A and B. The error function $\mu()$ introduces the factor $(1 - 2\epsilon)^{|\beta|}$ that effectively allows us to ignore βs with large size. Applying Cauchy-

Schwartz and Parseval's identity, we get Equation (2.4). Finally, we define a randomized labeling that shows consistency between A and B.

Overall, in addition to some intuitive ideas, one also needs playing around with Fourier coefficients and a trial-and-error approach. One needs to go back and forth between the definition of functions f, g, h and the resulting Fourier expressions, and finally arrive at the right test. Sometimes, one also needs the outer Label Cover instance to have some special properties and such instances must be constructed (e.g multi-layered smooth label cover in Theorem 4.2.4). It is also helpful to come up with counter-examples or cheating proof strategies. We cite a couple of such counter-examples below.

Note that α, β picked in the randomized labeling are guaranteed to be non-empty sets. This is because folding ensures (Lemma 2.4.4) that the coefficients $\widehat{A}_\emptyset, \widehat{B}_\emptyset$ are zero. However, folding is not only a clever trick, it is necessary. The verifier reads 3 bits x, y, z and accepts if $xyz = 1$. If one sets all bits in the proof to 1, then the test would always accept. Folding ensures that $A(-f) = -A(f)$. Hence the test of the verifier is actually $x'y'z' = 1$ where x', y', z' are either variables or their negations. Similarly, for showing hardness of MAX-3SAT, one designs a test with predicate $x \vee y \vee z$. But because of folding, the test is actually $x' \vee y' \vee z'$ where x', y', z' are positive or negative literals. In some applications like hypergraph coloring however, we cannot use folding. The reason is that folding introduces constraints saying "if vertex x is colored blue, then vertex x' must be colored red" ; such constraints cannot be enforced for coloring problems. In fact, inability to use folding turns out to be the biggest difficulty in showing hardness results for hypergraph coloring.

The use of error function $\mu()$ is another subtlety worth noting. Technically speaking, it allows us to ignore contribution of βs with large size. But at a deeper level, something like this is necessary. The error function $\mu()$ results in imperfect completeness and the resulting system of linear equations has no assignment that satisfies *all* equations. One can always find a satisfying assignment (by Gaussian elimination) in polynomial time if one exists. Hence imperfect completeness is necessary for NP-hardness results for linear predicates. In Chapter 3, we design a non-linear 5-bit test with perfect completeness.

In order to apply Parseval' identity, namely $\sum_\beta \widehat{B}_\beta^2 = 1$, one needs the coefficient $\widehat{B}_\beta$ in a squared form. This is achieved by reading *two* bits from table B. At least one bit must be read from table A, and therefore the test reads three bits. Let us see why we cannot build a 2-bit test. It would require us to read one bit from A and one bit from B which gives instead of Equation (2.3), something like $\sum_{\alpha,\beta:\alpha=\pi_2(\beta)} \widehat{A}_\alpha \widehat{B}_\beta$. We cannot apply Parseval, and in fact, this sum could potentially have large negative value. Thus we do not know how to analyze PCPs with only two queries; Fourier methods seem to fail. This is also a reason why we cannot prove good hardness results for vertex cover and coloring of graphs. We can however construct PCPs with three or more queries and show corresponding results for hypergraphs. Instead of just using Parseval, one could potentially use deeper results from Fourier analysis. In Chapter 8 and 9, we use Bourgain's Theorem and Friedgut's Theorem and construct 2-bit PCPs that yield hardness results for vertex cover in graphs and Min-2SAT-Deletion. However, these PCPs rely on a certain conjecture and it is an open problem to construct them unconditionally.

2.7 Techniques in this Thesis

In this section, we briefly mention some of the techniques developed in the thesis. We introduce new techniques at all levels of PCP design, namely, the Label Cover problem (or Outer Verifier), the encoding scheme and the PCP test (or Inner Verifier). There are several other problem-dependent techniques that we do not mention here.

Label Cover

The Label Cover instances given by Theorem 2.3.2 are not adequate for some applications, e.g. showing hardness of 3-uniform hypergraph coloring. The bipartite structure of the Label Cover instance turns out to be a bottleneck. The hypergraph constructed from such an instance also has bipartite structure and hence always 2-colorabale.

In Chapter 4, we build the state of the art outer verifier which we call Multi-layered Smooth Label Cover (Theorem 4.2.4). We use it to prove hardness of coloring 3-uniform hypergraphs (Chapter 4) and hardness of vertex cover in hypergraphs (Chapter 7). We believe that this verifier would have many applications in future.

In Chapter 8, we make a conjecture about existence of a new outer verifier. The conjecture has highly non-trivial implications, namely, optimal hardness results for Vertex Cover and Min-2SAT-Deletion.

Encoding Schemes

We make an extensive use of long code. Different tests for checking a long code are designed and analyzed using Fourier analysis. Checking whether a given string is a correct long code is same as checking whether a boolean function (given as a truth-table) depends on only one input coordinate. Therefore, theorems about Fourier spectrum of boolean functions have a natural place in PCP analysis. We use two such theorems, Theorem 8.3.2 due to Bourgain and Theorem 9.3.5 due to Friedgut.

In Chapters 4 and 5, we use long codes over non-boolean domains. This allows us to design some tests where analogous tests in the boolean case seem to fail.

For some applications however, long code turns out to be too inefficient in terms of its length. We instead use codes with much shorter length : in Chapter 10, we use Hadamard

Code and in Chapter 5, we introduce Split Code. We develop Fourier analysis techniques for these codes as well.

PCP Tests

We develop a variety of new tests depending on specific application. In Chapter 3, we use a 5-bit non-linear test and extend it in a query-efficient manner to a PCP with optimal amortized query complexity and perfect completeness. In Chapter 4 (Chapter 5 resp.), we use a test that reads three (four resp.) queries and accepts if and only if not all symbols are equal. This test is used to show hardness of coloring 3-uniform (4-uniform resp.) hypergraphs.

In Chapter 7 and 9, we use a combinatorial view of long code and show hardness results for (hyper)graph vertex cover. In the (hyper)graphs we construct, the (hyper)edges can be thought of as PCP tests. The constructions are analyzed using techniques from extremal combinatorics and Friedgut's Theorem.

Chapter 3

A Query Efficient PCP and Hardness of Graph Coloring

Graph Coloring is one of the most fundamental (and frustrating) problems in combinatorics and computer science. A graph is called K-colorable if every vertex can be assigned one color from a set of K colors so that endpoints of every edge receive different colors. Given a K-colorable graph for a constant $K \geq 3$, one seeks a polynomial time algorithm to color the graph using few colors. Best known algorithms use a huge number of colors, i.e. $n^{\Omega(1)}$ colors where n is the number of vertices in the graph. On the hardness side however, we only know that it is NP-hard to color 3-colorable graphs with 4 colors and K-colorable graphs with roughly $K^{3/2}$ colors for all large enough K.

In this chapter, we make a significant progress on this frustrating problem. We show that it is NP-hard to color K-colorable graphs with $K^{\Omega(\log K)}$ colors for all large enough K, obtaining the first superpolynomial lower bound for this problem (Theorem 3.1.1). It gives a strong evidence that perhaps it is NP-hard to color K-colorable graphs with *any* constant number of colors. The result is based on a PCP verifier that is very efficient in terms of the trade-off between the number of queries and the soundness parameter. The verifier achieves *perfect completeness*, a feature not shared by similar constructions known before. We also introduce a technique called *Randomized Label Cover* which could be useful to obtain further results on graph coloring.

3.1 Definitions, Results and Techniques

The main result in this chapter is :

Theorem 3.1.1 *[74] For sufficiently large constants K, it is NP-hard to color a K-colorable graph with $K^{\Omega(\log K)}$ colors. The result holds on graphs with degree $2^{K^{O(\log K)}}$.*

There is a huge gap between known algorithmic and hardness results for graph coloring. On the algorithmic side, Blum and Karger [18] give an algorithm to color a 3-colorable graph with $\tilde{O}(n^{3/14})$ colors whereas Karger et al [70] give an algorithm to color a K-colorable graph with $\tilde{O}(n^{1-3/(K+1)})$ colors. On the hardness side, Garey and Johnson [48] show that if for every K there exists an algorithm to color a K-colorable graph with $2K-6$ colors, then P = NP. But they do not specify an integer K for which it is NP-hard to color a K-colorable graph with $2K-6$ colors. Lund and Yannakakis [93] show that for every constant C, there exists a constant $K(C)$ such that it is NP-hard to color a $K(C)$-colorable graph with $C \cdot K(C)$ colors. Khanna et al [72] show that for every $K \geq 3$, it is NP-hard to color a K-colorable graph with $K' = K + 2\lfloor \frac{K}{3} \rfloor - 1$ colors. Fürer [47] shows the following result which was so far asymptotically the best result (definition of *amortized free bit complexity* appears next).

Theorem 3.1.2 *If NP has a PCP verifier with logarithmic randomness and amortized free bit complexity $\overline{f}$, then for every constant $\epsilon > 0$, for all sufficiently large constants*

K, it is hard to color a K-colorable graph with $K' = K^{1+\frac{1}{max(1+2\overline{f},2)}-\epsilon}$ colors assuming NP $\neq$ ZPP.

Using current PCPs with arbitrarily low amortized free bit complexity, this theorem gives $K' = K^{3/2-\epsilon}$. It was an open problem (also raised by Khanna et al [72]) whether one can replace K' by a superpolynomial function of K. Theorem 3.1.1 resolves this question. Our result is based on construction of a new verifier (Theorem 3.1.5) that is efficient in terms of its *amortized query complexity* and *amortized free bit complexity*.

Definition 3.1.3 *A verifier is said to have free bit complexity f if there is a subset of f queries read by the verifier such that the answer to every other query is determined by the answers to this subset of queries (if the verifier is to accept).*

For example, Håstad's 3-bit verifier reads 3 bits (x, y, z) and accepts if and only if $x \oplus y \oplus z = 0$ (or $x \oplus y \oplus z = 1$). This verifier has free bit complexity 2 since the answers for bits x, y automatically fix the answer to the bit z if the verifier is to accept.

If a verifier queries q bits of which f are free and has soundness s, then the *amortized query complexity* $\overline{q}$ and the *amortized free bit complexity* $\overline{f}$ are defined as

$$\overline{q} = \frac{q}{\log(1/s)}, \qquad \overline{f} = \frac{f}{\log(1/s)}$$

The amortized free bit complexity is an important parameter for showing hardness of approximating clique (or equivalently independent set). The reason is that in the fundamental connection between PCPs and inapproximability of clique (see [38]), the size of the FGLSS graph produced grows with parameter f. The soundness parameter on the other hand gives a "gap" between the clique sizes in completeness and soundness case. Thus the trade-off between f and s turns out to be crucial. The following result appears in [16].

Theorem 3.1.4 *If* NP *has a PCP verifier that uses logarithmic randomness, has completeness* $\geq \frac{1}{2}$ *and amortized free bit complexity* $\overline{f}$*, then assuming* $\mathrm{NP} \not\subseteq \mathrm{BPP}$*, no polynomial time algorithm can approximate clique size in an* n*-vertex graph within factor* $n^{\frac{1}{1+\overline{f}}-\epsilon}$*. Here* $\epsilon > 0$ *is an arbitrarily small constant.*

In this chapter, we construct the following PCP verifier that appears in a paper of Håstad and Khot [62].

Theorem 3.1.5 *For every integer* k*,* NP *has a PCP verifier that queries* $4k + k^2$ *bits of which* $4k$ *bits are free, has perfect completeness and soundness at most* 2^{-k^2+1}.

Theorem 3.1.5 gives a verifier that has $1 + \delta$ amortized query complexity and δ amortized free bit complexity for an arbitrarily small δ. The verifier is optimal in both the parameters and in particular we get an alternate proof of Håstad's [59] $n^{1-\epsilon}$ hardness result for Clique. Such a verifier was already constructed by Samorodnitsky and Trevisan [106], but their verifier loses perfect completeness and this seems to be essential for their proof. For many reasons it is preferable to have perfect completeness. Firstly, it is natural to have a proof system where a correct proof of a correct theorem is always accepted. Secondly, perfect completeness is sometimes essential to obtain further results. Some inapproximability results for problems such as coloring often make essential use of perfect completeness and when using a given PCP as a submodule in future PCPs, perfect completeness, to say the least, simplifies matters.

Several results in the past have focused on achieving PCPs with perfect completeness and this task often turns out to be much harder than obtaining corresponding PCPs without this property. For instance, Håstad shows that MAX-3SAT is hard to approximate within ratio $\frac{8}{7} - \epsilon$. This result follows from his 3-Bit PCP construction (Theorem 2.0.1) using a simple gadget that reduces Max-3-Lin-2 to MAX-3SAT. To extend this result to satisfiable instances however requires a new construction and a technically much more complicated proof.

Our result is based on a basic non-linear test which reads 5 bits $(b_1, b_2, b_3, b_4, b_5)$ from the proof and accepts if $b_1 = b_2 \oplus b_3 \oplus (b_4 \wedge b_5)$. We would like to emphasize that PCP with a linear predicate cannot achieve perfect completeness. We call this constraint Tri-Sum-And and let MAX-TSA be the problem of satisfying maximum number of such constraints. We have the following theorem.

Theorem 3.1.6 *For any $\epsilon > 0$, it is NP-hard to distinguish satisfiable instances of Max-TSA from those where one can satisfy only a fraction $\frac{1}{2} + \epsilon$ of the constraints.*

Note that this is tight for MAX-TSA since a random assignment satisfies half the constraints. We then iterate this basic test in a way similar to Samorodnitsky and Trevisan iterate the basic 3-bit test by Håstad. The iterated test suffices to prove Theorem 3.1.5. A standard reduction from PCPs to constraint satisfaction problems implies the following theorem :

Theorem 3.1.7 *Boolean constraint satisfaction problem on k variables is hard to approximate within ratio $2^{k-O(\sqrt{k})}$ on satisfiable instances.*

Techniques

We prove Theorem 3.1.6 by designing an appropriate PCP verifier that reads 5 bits from the proof. Denote this test by MAX-TSA$(b_1, b_2, b_3, b_4, b_5)$ which accepts if and only if $b_1 = b_2 \oplus b_3 \oplus (b_4 \wedge b_5)$. Think of b_2, b_3, b_4, b_5 as free bits and b_1 as depending on them. The test has perfect completeness and soundness essentially $\frac{1}{2}$. This test is then iterated in a query efficient way to prove Theorem 3.1.5. The iterated test first reads a set of $4k$ free bits, say S. Then it reads k^2 more bits, say $\{b_1^i | 1 \leq i \leq k^2\}$ and conducts the basic test MAX-TSA($b_1^i, b_2^i, b_3^i, b_4^i, b_5^i$) where $b_2^i, b_3^i, b_4^i, b_5^i \in S$ are the free bits already read. It accepts if and only if all k^2 basic tests accept. A very surprising result of Samorodnitsky and Trevisan [106] shows that these k^2 tests behave as independent tests, each test reduces the soundness by a factor $\frac{1}{2}$ and thus the iterated test has soundness essentially 2^{-k^2}.

However the analysis of both the basic test and the iterated test are rather technical. The result on graph coloring is proved using the iterated test as a black-box, so one could read that result without going through cumbersome analysis of PCP tests.

The graph coloring result is shown by invoking the following fundamental connection between PCPs and independent sets in graphs by Feige et al [38] : If there is a PCP that has near-perfect completeness, f free bits and soundness s, then there is a reduction from this PCP to a graph (called FGLSS graph) such that (i) in completeness case, the graph contains an independent set of (fractional) size 2^{-f} and (ii) in soundness case, there is no independent set of size s. Using the PCP of Theorem 3.1.5, we see that in the soundness case, the FGLSS graph has no independent set of size 2^{-k^2+1} and therefore needs 2^{k^2-1} colors to color it. On the other hand, in completeness case, we have a large independent set (of size 2^{-4k}). We might hope to have $2^{O(k)}$ such independent sets that cover the whole graph so that the graph is $2^{O(k)}$-colorable. An independent set in FGLSS graph corresponds to a proof for PCP, which in turn corresponds to a labeling to Label Cover instance. Therefore we need to ensure that the underlying Label Cover instance has *many* correct labelings. A simple technique called *Randomized Label Cover* allows us to achieve this (see Section 3.4.2).

Here is a simple example that motivates the randomization technique. Let us say we have a satisfiable SAT formula ϕ. Now we add some dummy variables to ϕ and construct a new formula ψ, keeping exactly the same clauses. Now ϕ has (at least) one satisfying assignment but ψ has many satisfying assignments since the dummy variables of ψ can be assigned arbitrary values. This (trivial) construction is the essence of the randomization technique. We have to apply a similar construction to the Label Cover problem. Things get a bit complicated since we have to look at the iterated test, combine it with the randomization technique and the FGLSS graph. We also need to prune a small portion of the FGLSS graph.

Graphs Vs Hypergraphs

In subsequent chapters, we will see strong hardness results for coloring hypergraphs. These results are shown by a completely different approach. To show hardness of coloring q-uniform hypergraphs, we construct a PCP that reads q symbols from the proof and accepts if not all symbols are equal. This is a q-query PCP with a specific acceptance predicate. Then we let the symbols (locations) in the proof to be vertices of a hypergraph and the tests of the verifier as the edges of the hypergraph. For $q \geq 3$, Fourier methods can be applied and the PCP can be analyzed in a rather straightforward way.

However, graphs correspond to the case $q = 2$, i.e. PCPs with two queries. Currently we do not know how to analyze 2-query PCPs with Fourier methods. Thus the difference between graphs and hypergraphs is rooted in the power of PCPs with two or more queries. The only connection we know between PCPs and independent sets in graphs is the FGLSS connection.

3.2 Basic test

In this section, we describe our basic 5-bit test with the acceptance predicate as MAX-TSA. Let $\mathcal{L} = (G(V, W, E), N, M, \{\pi^{v,w}\})$ be an instance of the Label Cover problem given by Theorem 2.3.2. Recall that $|N| = 2^u$, $|M| = 7^u$ and the soundness is $2^{-\gamma u}$. For bits $a, b \in \{-1, 1\}$, let $a \wedge b$ denote the logical AND of a, b where -1 represents logical TRUE and 1 represents logical FALSE.

The action of the verifier V_{5bit} is :

1. Pick a random vertex $w \in W$ and its random neighbor $v \in V$. Let $\pi = \pi^{v,w} : M \mapsto N$ be the corresponding projection function.

2. Let A, B be the supposed long codes of labels of v, w in the proof. These long codes are assumed to be folded (i.e. $A(-f) = -A(f)$).

3. Choose two random functions $g, g' \in \mathcal{G}$ and and two random functions $f, f' \in \mathcal{F}$ where

$$\mathcal{G} = \{g \mid g : M \mapsto \{-1, 1\}\}, \qquad \mathcal{F} = \{f \mid f : N \mapsto \{-1, 1\}\}$$

4. Define a function $h \in \mathcal{G}$ as $h = g(f \circ \pi)(g' \wedge (f' \circ \pi))$, i.e. by setting for each $y \in M$,

$$h(y) = g(y)f(\pi(y))(g'(y) \wedge f'(\pi(y)))$$

5. Accept if and only if

$$B(h) = B(g)A(f)(B(g') \wedge A(f'))$$

We have the basic completeness lemma.

Lemma 3.2.1 *The completeness of the basic test is 1.*

Proof: In a correct proof, A, B are long codes of labels $a \in N, b \in M$ resp. with $\pi(b) = a$. Hence $B(h) = h(b)$ and similarly for the other involved functions. The completeness now follows from the definition of h. ■

The major work is involved in establishing the soundness.

Lemma 3.2.2 *If the verifier in the basic test accepts with probability* $(1+\delta)/2$ *then there exists a labeling for the Label Cover instance* $\mathcal{L}$ *that satisfies* $\delta^{O(1)}$ *fraction of edges, i.e.* $OPT(\mathcal{L}) \geq \delta^{O(1)}$*. In particular, the soundness of the verifier is at most* $(1+\delta)/2$ *provided the parameter* u *of the Label Cover problem in Theorem 2.3.2 satisfies* $2^{-\gamma u} < \delta^{O(1)}$.

Proof: The acceptance probability of the verifier is

$$\text{Pr[accept]} = E_{v,w,f,f',g,g'}[\frac{1 + B(h)B(g)A(f)(B(g') \wedge A(f'))}{2}]$$

The hypothesis of the lemma implies that

$$E_{v,w,f,f',g,g'}[B(h)B(g)A(f)(B(g') \wedge A(f'))] = \delta. \tag{3.1}$$

Fix v, w, f' and g' and let us study

$$E_{f,g}[B(h)B(g)A(f)].$$

Replacing each function by its Fourier expansion we see that this equals (we denote $f \circ \pi, f' \circ \pi$ by f, f' resp. whenever the meaning is clear)

$$\sum_{\beta_1,\beta_2,\alpha} \hat{B}_{\beta_1}\hat{B}_{\beta_2}\hat{A}_{\alpha}E_{f,g}[\chi_{\beta_1}(fg(f' \wedge g'))\chi_{\beta_2}(g)\chi_{\alpha}(f)].$$

The expectation over g is zero unless $\beta_1 = \beta_2 = \beta$ and expectation over f is zero unless $\pi_2(\beta) = \alpha$ (definitions of sets $\pi(\beta)$ and $\pi_2(\beta)$ appear before Lemma 2.4.5).

Thus the expression equals

$$\sum_{\beta} \hat{B}_{\beta}^2 \hat{A}_{\pi_2(\beta)} \chi_{\beta}(f' \wedge g').$$

Hence we need to analyze

$$E_{f',g'}[\chi_{\beta}(f' \wedge g')(B(g') \wedge A(f'))].$$

We have $a \wedge b = \frac{1}{2}(1 + a + b - ab)$ and using this we should analyze

$E[\chi_{\beta}(f' \wedge g')], E[\chi_{\beta}(f' \wedge g')B(g')], E[\chi_{\beta}(f' \wedge g')A(f')],$ and $E[\chi_{\beta}(f' \wedge g')B(g')A(f')]$.

Fix the value of f' and let $\beta' = \{y \mid y \in \beta \ \wedge \ f'(\pi(y)) = -1\}$. Observe that

$$E_{g'}[\chi_{\beta}(f' \wedge g')] = E_{g'}[\prod_{y \in \beta}(f'(\pi(y)) \wedge g'(y))] = E_{g'}[\prod_{y \in \beta'} g'(y)] \ (= 0 \quad \text{unless } \beta' = \emptyset)$$

Similarly, the third expected value is 0 unless $\beta' = \emptyset$ while the second and the fourth expected values equal $\hat{B}_{\beta'}$ and $\hat{B}_{\beta'}A(f')$, respectively. The probability, over the choice of f' that β' is empty is $2^{-|\pi(\beta)|}$. Cauchy-Schwartz inequality implies

$$\begin{aligned} E_{f'}\left[|\hat{B}_{\beta'}|\right] &= 2^{-|\pi(\beta)|} \sum_{\alpha \subseteq \pi(\beta)} |\hat{B}_{\beta \cap \pi^{-1}(\alpha)}| \\ &\leq 2^{-|\pi(\beta)|/2} \left(\sum_{\alpha \subseteq \pi(\beta)} \hat{B}_{\beta \cap \pi^{-1}(\alpha)}^2 \right)^{1/2} \leq 2^{-|\pi(\beta)|/2} \qquad (3.2) \end{aligned}$$

This implies that we get an overall upper bound on the left hand side of (3.1) as

$$E_{v,w}\left[\sum_{\beta} \hat{B}_{\beta}^{2}\,|\hat{A}_{\pi_2(\beta)}|\,(2^{-|\pi(\beta)|}+2^{-|\pi(\beta)|/2})\right] \leq E_{v,w}\left[\sum_{\beta} \hat{B}_{\beta}^{2}\,|\hat{A}_{\pi_2(\beta)}|\,2^{1-|\pi(\beta)|/2}\right] \quad (3.3)$$

and hence this expression is at least δ. We use this to establish a good labeling for the Label Cover instance. We first establish that some parts of the given sum are small. We have the following lemma from [60].

Lemma 3.2.3 *There is a constant $c > 0$ such that the Label Cover instance given by Theorem 2.3.2 has the following additional property : For every $w \in W$, and $\beta \subseteq M$, if v is a randomly chosen neighbor of w and $\pi = \pi^{v,w}$, then*

$$E_v[|\pi(\beta)|^{-1}] \leq |\beta|^{-c}$$

The value $c = \frac{1}{35}$ is acceptable.

Let $S_\delta = (4(6+2\log\delta^{-1})/\delta)^{1/c}$ and consider any β of size at least S_δ. Since $E[|\pi(\beta)|^{-1}] \leq \delta/(4(6+2\log\delta^{-1}))$, we conclude that the probability that $|\pi(\beta)| \leq (6+2\log\delta^{-1})$ is bounded by $\delta/4$. Thus for any such β we have

$$E_v[2^{1-|\pi(\beta)|/2}] \leq \frac{\delta}{4}+\frac{\delta}{4}=\frac{\delta}{2}$$

and hence discarding terms with β of size at least S_δ in (3.3) still keeps a sum of expected value at least $\delta/2$.

Furthermore since $\sum_\beta \hat{B}_\beta^2 = 1$ we can discard any term with $|\hat{A}_{\pi_2(\beta)}| \leq \delta/4$ and not reduce the sum by more than $\delta/4$. We conclude that the sum which is the right hand side of (3.3) is at least $\delta/4$ even if we restrict summation to β of size at most S_δ and such that $|\hat{A}_{\pi_2(\beta)}| \geq \delta/4$.

Now consider the following randomized labeling for vertices in V, W of the Label Cover problem. For every $w \in W$, let B be the supposed long code of label of w. Choose $\beta \subseteq M$ with probability $\hat{B}_\beta^2$, pick a random $y \in \beta$ and define label$(w) = y$. Similarly for every $v \in V$, let A be the supposed long code of label of v. Choose $\alpha \subseteq N$ with probability $\hat{A}_\alpha^2$, pick a random $x \in \alpha$ and define label$(v) = x$. We note that since A, B are folded, by Lemma 2.4.4, the sets α and β selected by this procedure are nonempty. The fraction of edges satisfied by this labeling is at least

$$E_{v,w}\left[\sum_\beta \hat{B}_\beta^2 \hat{A}_{\pi_2(\beta)}^2 |\beta|^{-1}\right] \tag{3.4}$$

If we restrict summation to $|\beta| \leq S_\delta$ and $|\hat{A}_{\pi_2(\beta)}| \geq \delta/4$, expression (3.4) is at least

$$S_\delta^{-1}\delta/4 \; E_{v,w}\left[\sum_{\beta;|\beta|\leq S_\delta, |\hat{A}_{\pi_2(\beta)}|\geq \delta/4} \hat{B}_\beta^2 |\hat{A}_{\pi_2(\beta)}|\right]$$

By the above reasoning, this expected value is at least $\delta/4$ and we get a lower bound $S_\delta^{-1}(\delta/4)^2$ on $OPT(\mathcal{L})$. This completes the proof of the lemma. ■

The basic test reads 5 bits $(b_1, b_2, b_3, b_4, b_5)$ from the proof and checks whether $b_1 b_2 b_3 (b_4 \wedge b_5) = 1$ which is same as $b_1 = b_2 \oplus b_3 \oplus (b_4 \wedge b_5)$ in $\{0, 1\}$ notation. Theorem 3.1.6 now follows by a standard procedure of replacing the bits in the proof by variables and asking for a proof that maximizes the acceptance probability.

3.3 The Iterated Test (Almost Disjoint Sets Test)

Now we prove Theorem 3.1.5. We extend our basic test in a query efficient way. The verifier which we call V_{effi} is given a Label Cover instance $\mathcal{L}(G(V, W, E), N, M, \{\pi^{v,w}\})$.

The action of the verifier is :

- Pick $v \in V$ and pick k functions $(f_i)_{i=1}^k$ and k functions $(f'_j)_{j=1}^k$ all from $\mathcal{F}$.
- Pick k neighbors of v from W, say $(w_l)_{l=1}^k$, and k pairs of functions (g_l, g'_l) from $\mathcal{G}$.
- Perform the basic test for (f_i, f'_j, g_l, g'_l) for all triples (i, j, l) for which $i + j + l = 0 \bmod k$. Specifically, let

  $$h_{ijl} = g_l f_i (f'_j \wedge g'_l)$$

  and test if

  $$B_l(h_{ijl}) B_l(g_l) A(f_i)(A(f'_j) \wedge B_l(g'_l)) = 1 \tag{3.5}$$

Remark : Note that one could potentially carry out the test for all triples (i, j, l), but we do it only for k^2 triples (i, j, l) with $i + j + l = 0 \bmod k$. This set of triples has the property that any two triples intersect in at most one point (hence the name *almost disjoint sets test*). This property is crucial as we want to show that these k^2 tests behave as if they were independent tests.

The following theorem proves Theorem 3.1.5. The iterated test is analyzed using Håstad and Wigderson's [64] method which gives a simpler analysis of Samorodnitsky and Trevisan's PCP [106].

Theorem 3.3.1 *The iterated test has completeness* 1 *and soundness* $2^{-k^2} + \delta$ *provided* $2^{-\gamma u} < \delta^{O(1)}$. *In particular, the soundness is* 2^{-k^2+1} *provided* $u = O(k^2)$. *The test queries* $4k + k^2$ *bits of which* $4k$ *are free.*

Proof of Theorem 3.3.1

The completeness follows from that of the basic test and we need to analyze the soundness. Let Z_0 denote the set of triples (i, j, l) with $i + j + l = 0 \bmod k$. Let $\mathrm{Acc}(i, j, l)$ be a variable that indicates whether the test given by the triple (i, j, l) accepts, taking the value 1 if it does and -1 otherwise. Note that in fact

$$\mathrm{Acc}(i, j, l) = B_l(h_{ijl})B_l(g_l)A(f_i)(A(f'_j) \wedge B_l(g'_l))$$

Consider

$$\prod_{(i,j,l)\in Z_0} \frac{1 + \mathrm{Acc}(i, j, l)}{2} = 2^{-k^2} \sum_{S\subseteq Z_0} \prod_{(i,j,l)\in S} \mathrm{Acc}(i, j, l) \tag{3.6}$$

This number equals 1 if the test accepts and is 0 otherwise and thus its expected value is the probability that the test accepts. The expectation is over the choice of v, $(f_i)_{i=1}^k$, $(f'_j)_{j=1}^k$, $(w_l, g_l, g'_l)_{l=1}^k$.

The term with $S = \emptyset$ is 1 and to establish the theorem it is sufficient to establish that any other term is bounded by δ. Let T_S be the expected value of the term corresponding to S. We go on to define a labeling for the Label Cover instance which satisfies a fraction $|T_S|^{O(1)}$ of edges.

Suppose that $(i_0, j_0, l_0) \in S$ and let us fix the values of f_i, $i \neq i_0$, f'_j, $j \neq j_0$ and (w_l, g_l, g'_l) for $l \neq l_0$ in such a way as not to decrease $|T_S|$. Consider any triple $(i, j, l) \in S$ other than (i_0, j_0, l_0). It intersects (i_0, j_0, l_0) in at most one place. If $i \neq i_0, j \neq j_0, l \neq l_0$, after the fixings, $\mathrm{Acc}(i, j, l)$ reduces to a constant ± 1. Similarly, if $i = i_0, j \neq j_0, l \neq l_0$, $\mathrm{Acc}(i, j, l)$ reduces to $X(f_{i_0})$ where X is some function of f_{i_0}. An important point is that X depends only on the choice of v. If $i \neq i_0, j = j_0, l \neq l_0$, $\mathrm{Acc}(i, j, l)$ reduces to some function $Y(f'_{j_0})$ depending only on v. Finally if $i \neq i_0, j \neq j_0, l = l_0$, Acc$(i, j, l)$ reduces to some function $Z(g_{l_0}, g'_{l_0})$ depending both on w_{l_0} and v.

On the other hand, using $x \wedge y = \frac{1+x+y-xy}{2}$ one can write

$$\text{Acc}(i_0, j_0, l_0) = B_{l_0}(h_{i_0 j_0 l_0}) B_{l_0}(g_{l_0}) A(f_{i_0}) \cdot \frac{1 + A(f'_{j_0}) + B_{l_0}(g'_{l_0}) - A(f'_{j_0}) B_{l_0}(g'_{l_0})}{2}$$

Altogether we can write T_S as a sum of 4 terms of the form

$$B_{l_0}(h_{i_0 j_0 l_0}) A'(f_{i_0}) A''(f'_{j_0}) C(g_{l_0}, g'_{l_0}) \tag{3.7}$$

each with a coefficient $1/2$. Here A', A'', C are boolean functions that "absorb" all the functions of type X, Y, Z respectively. We again stress that A' and A'' only depend on v and hence can be used to define a labeling for v. B_{l_0} is the original long code for w_{l_0} and hence is useful to define a labeling for w_{l_0}. Finally C is a Boolean function that depends on both v and w_{l_0} and is not useful to define a labeling.

Let us now discard the indices for readability and write (3.7) as

$$B(h) A'(f) A''(f') C(g, g')$$

We want to compute the expected value of this expression over random choices of f, f', g and g'. Expanding all factors except $A''(f')$ by the Fourier transform we get

$$\sum_{\alpha,\beta,\gamma,\gamma'} \hat{A}'_{\alpha} \hat{B}_{\beta} \hat{C}_{\gamma,\gamma'} E[\chi_{\alpha}(f) \chi_{\beta}(g f (f' \wedge g')) \chi_{\gamma}(g) \chi_{\gamma'}(g') A''(f')] \tag{3.8}$$

Now taking the expected value over f we see that unless $\alpha = \pi_2(\beta)$ the term is zero. Similarly we need $\beta = \gamma$, and fixing f' we see that unless $\gamma' = \beta \cap \pi^{-1}(f'^{-1}(-1))$ we also get a zero expected value. Thus the expression reduces to

$$\sum_{\beta} \hat{A}'_{\pi_2(\beta)} \hat{B}_{\beta} \hat{C}_{\beta,\beta\cap\pi^{-1}(f'^{-1}(-1))} A''(f'). \tag{3.9}$$

We have

$$\begin{aligned} |\, E_{f'}[\hat{C}_{\beta,\beta\cap\pi^{-1}(f'^{-1}(-1))} A''(f')] \,| \quad &\leq \quad E_{f'}[\, |\hat{C}_{\beta,\beta\cap\pi^{-1}(f^{-1}(-1))}| \,] \\ &\leq \quad 2^{-|\pi(\beta)|} \sum_{\alpha'\subseteq\pi(\beta)} |\hat{C}_{\beta,\beta\cap\pi^{-1}(\alpha')}| \\ &\leq \quad 2^{-|\pi(\beta)|/2} \left(\sum_{\alpha'\subseteq\pi(\beta)} \hat{C}^2_{\beta,\beta\cap\pi^{-1}(\alpha')} \right)^{1/2} \end{aligned}$$

Substituting this estimate into (3.9) and using Cauchy-Schwartz inequality over β we get the upper estimate

$$\left(\sum_{\beta} \hat{B}^2_{\beta} \hat{A}'^2_{\pi_2(\beta)} 2^{-|\pi(\beta)|} \right)^{1/2} \left(\sum_{\beta,\beta_1} \hat{C}^2_{\beta,\beta_1} \right)^{1/2} \leq \left(\sum_{\beta} \hat{B}^2_{\beta} \hat{A}'^2_{\pi_2(\beta)} 2^{-|\pi(\beta)|} \right)^{1/2}$$

for $|T_S|$. The rest of the proof now follows along the same lines as for the basic test. We define the same labeling and the analysis is almost identical. We omit the details.

3.4 Hardness of Graph Coloring

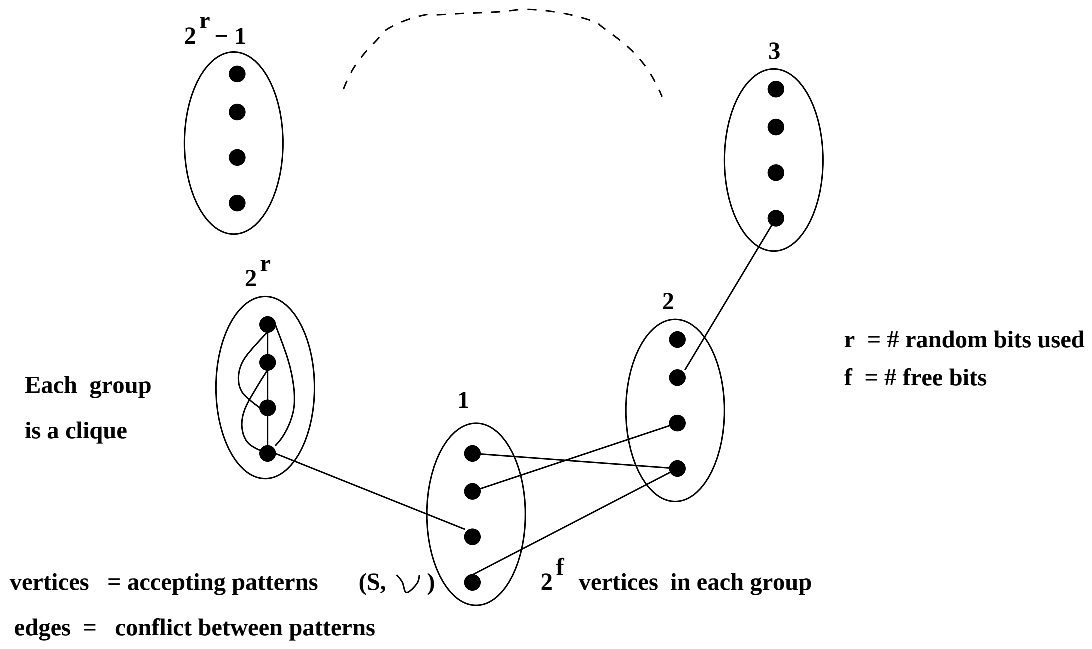
We prove Theorem 3.1.1 in this section. We give a reduction from the PCP verifier V_{effi} in Section 3.3 to a graph G such that : in completeness case, the graph can be colored with K colors and in soundness case, the graph needs $K^{\Omega(\log K)}$ colors to color it. We use a modification of the verifier V_{effi} and use Theorem 3.3.1. We also use the fundamental connection between PCPs and independent sets in graphs, discovered by Feige et al [38]. We first describe their construction that builds a graph from any PCP verifier.

Definition 3.4.1 *An accepting pattern τ for a PCP verifier is a pair $\tau = (S, \nu)$ such that for some choice of the random string, S is the set of query bits read by the verifier and ν is a setting of these bits for which the verifier would accept. The set of all accepting patterns is denoted by $\mathcal{T}$. A proof Π is said to be consistent with a pattern $\tau = (S, \nu)$ if the values of bits in proof Π corresponding to the set S match ν.*

3.4.1 The FGLSS Graph

Given a PCP verifier V_{pcp} that uses r random bits and f free bits, we define the corresponding FGLSS graph G as follows (Fig. 3.1). The vertices of this graph are all accepting patterns $\tau = (S, \nu)$. There is an edge between two patterns (S, ν) and (S', ν') if the sets S, S' have a bit in common and ν, ν' assign different values to this bit, i.e. if these patterns are conflicting. Note that there is one set S of queries for every choice of the random string used by the verifier and there are 2^f settings ν such that (S, ν) is an accepting pattern. So the FGLSS graph has $|G| = 2^{r+f}$ vertices. It is convenient to group the vertices into 2^r groups, one group for every random string. Each group contains 2^f vertices which form a clique.

The important property of the FGLSS graph is the following : For a proof Π, consider the set of patterns that are consistent with this proof. This set consists of one pattern for every random string on which the verifier accepts. Since these patterns are consistent with the proof Π, they are non-conflicting and hence form an independent set in the FGLSS



$2^r - 1$
3
2^r
2
r = # random bits used
f = # free bits
Each group
is a clique
1
vertices = accepting patterns
(S, ∨)
2^f vertices in each group
edges = conflict between patterns

Figure 3.1: FGLSS Graph

graph. We call this independent set I_Π. Its size is given by

$$\begin{aligned} |I_\Pi| &= \sharp \text{ random strings for which proof } \Pi \text{ is accepted} \\ &= 2^r \cdot (\text{acceptance probability for proof } \Pi\ ) \end{aligned} \tag{3.10}$$

Conversely, an independent set in the FGLSS graph gives a proof whose acceptance probability is proportional to the size of this independent set, as given by the above equation. Thus there is a one-to-one correspondence between proofs for the verifier and independent sets in the FGLSS graph.

Now consider the verifier V_{effi} described in Section 3.3. Its soundness is at most 2^{-k^2+1}. Equation (3.10) implies that the size of a maximum independent set in the corresponding FGLSS graph is at most $2^r \cdot 2^{-k^2+1}$. Since every independent set is "small", the graph needs lots of colors to color it. Specifically, we need at least

$$\frac{|G|}{2^r \cdot 2^{-k^2+1}} = \frac{2^{r+f}}{2^r \cdot 2^{-k^2+1}} = \frac{2^{r+4k}}{2^r \cdot 2^{-k^2+1}} = 2^{k^2+4k-1}$$

colors to color the FGLSS graph.

We investigate the completeness case next. We would like to show that the graph can be colored with a small number of colors, or equivalently it can be covered by a small number of independent sets. This is not necessarily true for the FGLSS graph constructed from the verifier V_{effi}, so we need to construct a new verifier which we call V_{rand}.

Note that an independent set in FGLSS graph corresponds to a correct proof, which in turn, corresponds to a correct labeling of the Label Cover instance $\mathcal{L}$. Thus, in order to have many independent sets in the FGLSS graph, we need to have many correct labelings to the Label Cover instance. The following construction gives a new instance $\mathcal{L}'$ of Label Cover from the original instance $\mathcal{L}$ with the extra property that it has many correct labelings.

3.4.2 Randomized Label Cover

Definition 3.4.2 *Given a Label Cover instance $\mathcal{L} = (G(V,W,E), N, M, \{\pi^{v,w}\})$, and a finite set Z, the randomized Label Cover instance $\mathcal{L}' = (G(V,W,E), N', M', \{\pi'^{v,w}\})$ is defined as follows.*

- $M' = M \times Z, \;\; N' = N \times Z$
- *For $(v,w) \in E$, and the projection map $\pi^{v,w} : M \mapsto N$, the new projection map $\pi'^{v,w} : M' \mapsto N'$ is defined as*

$$\pi'^{v,w}((b,z)) = (\pi^{v,w}(b), z) \quad \forall\,(b,z) \in M' = M \times Z$$

To summarize, $\mathcal{L}'$ has the same set of vertices and edges as $\mathcal{L}$. However, the label sets are now $M \times Z$ and $N \times Z$. The new projection maps are same as the previous projection maps on the first coordinate and identity on the second coordinate. Thus the second coordinate is "dummy". The following properties are obvious.

Lemma 3.4.3 *If $\mathcal{L}'$ is randomized Label Cover instance obtained from $\mathcal{L}$ and a set Z, then*

- *OPT($\mathcal{L}'$) = OPT($\mathcal{L}$).*
- *If a labeling $\Phi : V \mapsto N, \Phi : W \mapsto M$ is a correct labeling for $\mathcal{L}$ (i.e. labeling that satisfies every edge), then for every $z_0 \in Z$, the labeling $\Phi' : V \mapsto N', \Phi' : W \mapsto M'$ is a correct labeling for $\mathcal{L}'$. The labeling Φ' is defined as*

  $$\Phi'(v) = (\Phi(v), z_0), \;\; \Phi'(w) = (\Phi(w), z_0)$$

  In particular, a correct labeling of $\mathcal{L}$ gives $|Z|$ correct labelings of $\mathcal{L}'$ for different choices of z_0.

3.4.3 Pruning FGLSS Graph and Proof of Theorem 3.1.1

The idea is to construct a new verifier V_{rand} that works as follows : Given a Label Cover instance $\mathcal{L}$, first construct a randomized Label Cover instance $\mathcal{L}'$ and then run the verifier V_{effi} (from Section 3.3) on $\mathcal{L}'$. We choose $|Z| = 2^{5k}$.

Recall that in the completeness case, there are $|Z|$ different labelings to $\mathcal{L}'$ and hence there are several correct proofs (i.e. proofs accepted with probability 1), one proof $\Pi(z)$ for every choice of $z \in Z$. Each of these correct proofs corresponds to an independent set of size 2^r. We may expect that these independent sets cover the FGLSS graph. This essentially turns out to be the case : they cover all but a tiny portion of the FGLSS graph. This tiny portion can be identified beforehand and thrown away. Thus the remaining FGLSS graph can be colored with a small number (2^{5k}) of colors. The next definition helps us identify this "bad" portion of the FGLSS graph. Recall that the vertices of the FGLSS graph correspond to the verifier's choice of the tables $(A, B_1, \dots, B_k)$ and the functions $(f_i, f'_j, g_l, g'_l)_{i,j,l=1,2,\dots,k}$. The functions f_i, f'_j are now on the domain $N' = N \times Z$ and the functions g_l, g'_l are now on the domain $M' = M \times Z$.

We will identify some of the choices of the functions as "bad".

Definition 3.4.4 *A choice of functions* $(f_i, f'_j, g_l, g'_l)_{i,j,l=1,2,\dots,k}$ *is called good if*

$$\forall\, a \in N,\ \forall\, b_1, b_2, \dots, b_k \in M,\ \forall\, x \in \{-1,1\}^{4k},\ \exists\, z \in Z \textit{ such that}$$

$$x = (\ f_1(a,z), f'_1(a,z), \dots, f_k(a,z), f'_k(a,z), g_1(b_1,z), g'_1(b_1,z), \dots, g_k(b_k,z), g'_k(b_k,z)\ )$$

Lemma 3.4.5 *If* $|Z| = 2^{5k}$*, then the probability that a choice of functions* (f_i, f'_j, g_l, g'_l) *is not good is* $\le 2^{-2^{k-1}}$ *provided* k *is large enough.*

Proof: Fix $x, a, b_1, \dots, b_k$. Note that (f_i, f'_j, g_l, g'_l) are defined by setting value ± 1 with equal probability at every point independently. So for every $z \in Z$, the probability that

$$x \neq (\ f_1(a,z), f'_1(a,z), \dots, f_k(a,z), f'_k(a,z), g_1(b_1,z), g'_1(b_1,z), \dots, g_k(b_k,z), g'_k(b_k,z)\ )$$

is $1 - 2^{-4k}$. The probability that this holds for every $z \in Z$ is $(1 - 2^{-4k})^{2^{5k}} \le 2^{-2^k}$. Now we take a union bound over all choices of $x, a, (b_i)_{i=1}^k$. There are 2^{4k} choices for x, $|N| = 2^u$ choices for a, and $|M| = 7^u$ choices for each of $b_1, b_2, \dots b_k$. Thus the total number of choices is $2^{O(ku)}$ and $u = O(k^2)$ (see Theorem 3.3.1). It follows that a choice of functions is not good with probability at most $2^{-2^k + O(k^3)} \le 2^{-2^{k-1}}$. ■

We remove the vertices in the FGLSS graph which correspond to a bad choice of functions and call the remaining graph as the *modified FGLSS graph*. By Lemma 3.4.5, the fraction of vertices removed is very small. So in the soundness case, we still need at least 2^{k^2} colors to color the modified FGLSS graph.

In the completeness case, consider any vertex (S, ν) of the modified FGLSS graph where S corresponds to a set of queries ($\{A(f_i)\}_{i=1}^{k}$, $\{A(f'_j)\}_{j=1}^{k}$, $\{B_l(g_l), B_l(g'_l)\}_{l=1}^{k}$) and ν is some setting of these bits. In a correct proof $\Pi(z)$ for verifier V_{rand}, the tables A, B_j are long codes of some labels $(a, z), (b_j, z)$. Hence

$$A(f_i) = f_i(a, z),\ A(f'_j) = f'_j(a, z),\ B_l(g_l) = g_l(b_l, z),\ B_l(g'_l) = g'_l(b_l, z) \qquad (3.11)$$

In the modified FGLSS graph, we are guaranteed that for some choice of z, the bits in (3.11) match the bit-pattern ν (by Definition 3.4.4). Thus the independent sets corresponding to the proofs $\{\Pi(z)\}_{z \in Z}$ cover every vertex of the modified FGLSS graph and hence it can be colored with $|Z| = 2^{5k}$ colors.

Thus the modified FGLSS graph can either be colored with 2^{5k} colors or requires 2^{k^2} colors to color it. Taking $K = 2^{5k}$ proves Theorem 3.1.1. It is easy to see that the FGLSS graph has degree at most $2^{K^{O(\log K)}}$.

Chapter 4

Hardness of Coloring 3-Uniform Hypergraphs

Coloring 3-colorable graphs is one of the most important open problems in combinatorics and computer science. The current best algorithms ([18], [70]) require $\tilde{O}(n^{3/14})$ colors where n is the number of vertices in the graph. On the other hand, we only know that it is NP-hard to color 3-colorable graphs with 4 colors [72]. Current techniques seem to have stuck on this problem and therefore it is natural to study its generalization to hypergraphs.

In this chapter, we obtain first strong hardness result for coloring 3-uniform hypergraphs. We show that it is hard to color 3-colorable 3-uniform hypergraphs with $(\log\log n)^{\Omega(1)}$ colors. In terms of techniques, our main contribution is construction of a new PCP outer verifier, which we call Multi-layered Smooth Label Cover (Theorem 4.2.4). It is also used in Chapter 7 for showing hardness of Hypergraph Vertex Cover. This verifier represents the state of the art outer verifier and could have many other applications in future.

Hardness results were earlier known for coloring 4-uniform hypergraphs (see [55]) and we overcome the hurdles involved in extending these results to 3-uniform hypergraphs. Hardness results for graph coloring however seem out of reach of current techniques. We point out a fundamental difference between graphs and hypergraphs that sheds some light on the difficulty in attacking graph coloring problem.

4.1 Definitions, Results and Techniques

A q-uniform hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{E})$ consists of a set of vertices $\mathcal{V}$ and a set of edges $\mathcal{E}$. Every edge $e \in \mathcal{E}$ is a size-q subset of the set of vertices (so graphs are 2-uniform hypergraphs). A hypergraph is said to be k-colorable if the vertices can be colored with k colors so that for every edge, not all its vertices have the same color. We consider both, the minimization and maximization versions of hypergraph coloring problem.

Minimization Version : In the minimization version (also called Approximate Coloring problem), we are given a k-colorable hypergraph where k is a small constant and we seek an algorithm to color the hypergraph with as few colors as possible. It is well-known that it is NP-hard to test 3-colorability of graphs whereas Lovász [90] showed that it is NP-hard to test 2-colorability of 3-uniform hypergraphs. The best known algorithms for (hyper)graph coloring are summarized in Table 1. Explicit algorithms are known only for 2-colorable 3-uniform hypergraphs. However, it is a folklore result that for any $q \geq 3, k \geq 2$, one can color k-colorable q-uniform hypergraphs in polynomial time with $n^{c(q,k)}$ colors for some constant $c(q, k) < 1$.

A big open problem is whether 3-colorable graphs are hard to color with constantly many colors. Surprisingly, Guruswami et al [55] were able to show such a result for 4-uniform hypergraphs. They showed hardness of coloring 2-colorable 4-uniform hy-

	Holds for	**Colors with**
Blum, Karger [18]	3-colorable graphs	$\tilde{O}(n^{3/14})$ colors
Karger et al [70]	k-colorable graphs, $k \geq 3$	$\tilde{O}(n^{1-\frac{3}{k+1}})$ colors; Improvements by Halperin et al [58]
Krivelevich et al [81]	2-colorable 3-uniform hypergraphs	$\tilde{O}(n^{1/5})$ colors

Table 1 : Known algorithmic results for (hyper)graph coloring.

	Holds for	**Hardness of coloring with**	**Assumption**
Khanna et al [72]	3-colorable graphs	4 colors	P $\neq$ NP
Khot [74] Chapter 3	k-colorable graphs, all sufficiently large k	$k^{\Omega(\log k)}$ colors	P $\neq$ NP
Guruswami et al [55]	2-colorable 4-uniform hypergraphs	constantly many colors	P $\neq$ NP
Guruswami et al [55]	2-colorable 4-uniform hypergraphs	$\Omega(\frac{\log \log n}{\log \log \log n})$ colors	NP $\not\subseteq$ DTIME($n^{O(\log \log n)}$)
Khot [76] Chapter 5	k-colorable 4-uniform hypergraphs, $k \geq 5$	$(\log n)^{ck}$ colors, $c > 0$ absolute constant	NP $\not\subseteq$ DTIME($2^{(\log n)^{O(1)}}$)
Khot [75] This chapter	3-colorable 3-uniform hypergraphs	constantly many colors	P $\neq$ NP
Khot [75] This chapter	3-colorable 3-uniform hypergraphs	$(\log \log n)^{1/8}$ colors	NP $\not\subseteq$ DTIME($n^{O(\log \log n)}$)
Dinur et al [31]	2-colorable 3-uniform hypergraphs	constantly many colors	P $\neq$ NP
Dinur et al [31]	2-colorable 3-uniform hypergraphs	$(\log \log n)^{1/3}$ colors	NP $\not\subseteq$ DTIME($2^{(\log n)^{O(1)}}$)

Table 2 : Known hardness results for (hyper)graph coloring.

pergraphs with constantly many colors. A question left open by Guruswami et al [55] was whether similar hardness result holds for 3-uniform hypergraphs. We answer this question positively by proving that :

Theorem 4.1.1 *For every constant $\delta > 0$, it is NP-hard to distinguish whether an n-vertex 3-uniform hypergraph is 3-colorable or it contains no independent set of size δn. In particular, it is NP-hard to color 3-colorable 3-uniform hypergraphs with constantly many colors.*

Independent set in a hypergraph is defined as a set of vertices such that no edge lies entirely within this set. In a properly colored hypergraph, the vertices colored with the same color form an independent set. Thus saying that a hypergraph contains no independent set of size δn is stronger than saying it cannot be colored with $1/\delta$ colors. The above theorem implies that given a hypergraph that contains an independent set of size $\Omega(n)$ ($\frac{n}{3}$ in this case), it is hard to find an independent set of size δn for any constant $\delta > 0$. Such a result was first proved in [76] and [66] for 4-uniform hypergraphs. Obtaining a similar result for graphs is a major open problem and is equivalent to constructing PCPs with zero free bits, completeness $\Omega(1)$ and arbitrarily low soundness. There is no such PCP characterization for independent sets in hypergraphs. Thus one outcome of the work on hypergraph coloring is to point out the fundamental difference between graphs and hypergraphs. Guruswami et al's paper [55] gave hope that techniques for hypergraphs might eventually be used for graphs, but evidence from subsequent work has been negative.

Under a stronger complexity assumption than P $\neq$ NP, we show the following stronger hardness result.

Theorem 4.1.2 *Assuming $\mathrm{NP} \not\subseteq \mathrm{DTIME}(n^{O(\log\log n)})$, it is hard to color a 3-colorable 3-uniform hypergraph on n vertices with $(\log\log n)^{1/8}$ colors.*

Maximization Version : In the maximization version of the hypergraph coloring problem, we are given a q-uniform hypergraph and k different colors. The goal is to assign one color to every vertex so as to maximize the number of edges that are non-monochromatic.

One can also think of the maximization version as a constraint satisfaction problem (CSP) on the Not-All-Equal predicate. The vertices of the hypergraph are variables of the CSP and the edges are constraints of the CSP. Since the hypergraph is q-uniform, every constraint is defined on q variables. Assignment of colors to the vertices corresponds to assigning every variable a value from a domain of size k. A constraint is satisfied iff all its variables are *not* assigned the same value, or equivalently, the corresponding edge is non-monochromatic. We call this CSP to be the problem NAE $_{q,k}$. The optimum of the CSP is the maximum fraction of the constraints that can be satisfied by any assignment.

Note that assigning every variable a random value from the size-k domain satisfies $(1 - \frac{1}{k^{q-1}})$ fraction of the constraints. The problem NAE $_{q,k}$ is said to have a *Random Threshold Property* if it is NP-hard to do strictly better than assigning random values, or more specifically, if it is NP-hard to distinguish whether the optimum is $\geq 1 - \epsilon$ or $\leq 1 - \frac{1}{k^{q-1}} + \epsilon$ for arbitrarily small $\epsilon > 0$.

NAE$_{2,k}$: For graphs ($q = 2$), the case $k = 2$ corresponds to the MAX-CUT problem and Goemans and Williamson [50] give an algorithm that performs strictly better than taking a random cut. Frieze and Jerrum [46] extend this algorithm for $k \geq 3$ colors. Thus the problem NAE$_{2,k}$ *does not* have the random threshold property for any $k \geq 2$.

NAE$_{4,k}$: Håstad [60] showed that for the problem NAE $_{4,2}$, a gap $(1, \frac{7}{8} + \epsilon)$ is hard, i.e. this problem has the random threshold property. It is implicit in his work that NAE$_{4,k}$ has the random threshold property for every $k \geq 2$. This result was the basic starting point for the hypergraph coloring result of Guruswami et al [55].

NAE$_{3,k}$: Quite interestingly, Zwick [115] showed that for the problem NAE $_{3,2}$, there exists an algorithm that does strictly better than the random assignment. This seemed to be a stumbling block in extending Guruswami et al's techniques to 3-uniform hypergraphs. In this chapter, we resolve the only remaining case i.e. NAE$_{3,k}$ with $k \geq 3$. We show that

Theorem 4.1.3 *For every $k \geq 3$, the problem* NAE $_{3,k}$ *has the random threshold property on satisfiable instances. In particular, it is NP-hard to distinguish whether a 3-uniform hypergraph is 3-colorable or any coloring of the vertices with 3 colors has at most $\frac{8}{9} + \epsilon$ fraction of the edges non-monochromatic. Here $\epsilon > 0$ is an arbitrarily small constant.*

Equivalent Formulation in terms of PCPs

We prove Theorem 4.1.1 and 4.1.3 by constructing a suitable PCP. We construct a PCP verifier that expects a proof over the ternary domain $\mathbf{Z}_3$ (or k-ary domain $\mathbf{Z}_k$). The verifier reads 3 symbols from the proof and accepts if and only if not all 3 symbols are equal. We let the locations in the proof to be the vertices of a hypergraph and the tests of the verifier (reading 3 locations) to be the edges of the hypergraph. This defines a 3-uniform hypergraph. In completeness case, we show that there exists a proof that the verifier always accepts and hence the hypergraph is 3-colorable. In soundness case, we show that the hypergraph has no independent set of size δn. This proves Theorem 4.1.1. We also show in soundness case that the probability of acceptance of the verifier is at most $\frac{8}{9} + \epsilon$ which proves Theorem 4.1.3. For the latter result, we need to construct a somewhat different PCP so that it works for every $k \geq 3$.

Remark : Guruswami et al [55] use a technique called *covering complexity* to analyze their PCP. The hypergraph they construct indeed has no independent set of size δn, but this fact cannot be proven using covering complexity method. This was proved using a more direct approach by Holmerin [66].

Techniques

The main technique in this chapter is construction of a new version of Label Cover problem which we call Multi-layered Smooth Label Cover problem.

Motivation for multi-layered structure : The Label Cover problem defined in Theorem 2.3.2 has a bipartite structure, i.e. the underlying graph has two layers. This 2-layered structure turns out to be a bottle-neck in showing certain hardness of approximation results. Let us say we want to prove Theorem 4.1.1. As explained before, this is equivalent to building a 3-query PCP with Not-All-Equal predicate. Consider such a PCP built from a 2-layered Label Cover problem. In this PCP, the proof naturally splits into two parts. Verifiers that can be built using current techniques are forced to read one query from the left part and two queries from the right part. When a hypergraph is constructed from such a PCP, the hypergraph also splits into two layers and every hyperedge has one vertex in left layer and two vertices in the right layer. Coloring all vertices in each layer with the same color gives a proper 2-coloring of the hypergraph. Thus the hypergraph is always 2-colorable and the construction is doomed.

In the multi-layered version of Label Cover problem, the underlying graph has many layers and between every pair of layers, we have an instance of the usual Label Cover problem. A hypergraph built from a multi-layered version has many layers and there are hyperedges between every pair of layers. Thus the "cheating strategy" described above breaks down with the multi-layered construction.

Motivation for smoothness : Significance of smoothness property is rather technical in nature. Smoothness refers to the property of the maps $\pi^{v,w}$ in the definition of the Label Cover problem (Definition 2.3.1). In general, these maps are many-to-one maps. As we will see in Chapter 8, one possible direction for getting hardness results for some open problems (e.g. Min-2SAT-Deletion, Vertex Cover), is to show hardness of Label Cover instances with the property that the maps $\pi^{v,w}$ are bijections. In other words, we would like to have, for every edge (v, w), and every pair of distinct labels $b, b' \in M$,

$$\pi^{v,w}(b) \neq \pi^{v,w}(b')$$

A simple and powerful technique in this chapter is to get a weaker analogue of the above property. We will show hardness of Label Cover instances where the maps $\pi^{v,w}$ are "smooth". For every $w \in W$ and every pair of distinct labels $b, b' \in M$, we have

$$\text{Pr}_v[\pi^{v,w}(b) \neq \pi^{v,w}(b')] \approx 1$$

Thus over the choice of a random neighbor of w, the projections of labels b and b' are distinct with high probability. The significance of this property will be evident only when we get to the analysis. We build Label Cover instances with both the properties : multi-layered structure and smoothness.

Remark : The idea of multi-layered Label Cover is from Dinur et al [29] and the smoothness property is from Khot [75]. The latter paper combines the two ideas and that is what we present in this chapter.

Comparison with Dinur et al's Work :

Dinur, Regev and Smyth [31] obtained a better hardness result for coloring 3-uniform hypergraphs, and our work was partly influenced by their result. They show hardness of coloring 2-colorable 3-uniform hypergraphs with constantly many colors. However their construction has large independent sets (in fact independent sets of size $\frac{n}{2}$) and in this respect, their result is weaker than Theorem 4.1.1. Both results use the multi-layered Label Cover problem. We, in addition, use smoothness property and Fourier analysis whereas Dinur et al make a clever use of Kneser graphs and their construction is more combinatorial in nature. It is interesting that they are able to get a hardness result for 2-colorable 3-uniform hypergraphs even though the problem NAE $_{3,2}$ does *not* have the random threshold property. It would be nice, if possible, to obtain a Fourier analysis based proof of their result.

4.2 Constructing Multi-layered Smooth Label Cover

4.2.1 Achieving Smoothness Property

Let us first focus on achieving smoothness property. We will later present a combined construction that achieves both smoothness and multi-layered structure.

We modify the 2-Prover-1-Round game (i.e. Raz's Verifier) in Section 2.2 in the following way : The verifier is given an instance ϕ of Gap-3SAT-5. Let u, T be parameters. The verifier picks randomly a set w' of u clauses and a set v' of u variables as before; variables in v' include one variable from each clause in w'. Then he picks a set w'' of Tu clauses at random. He asks the second prover to give a satisfying assignment to the set

$w = w' \cup w''$ and asks the first prover to give a satisfying assignment to the set $v = v' \cup w''$. The verifier accepts if and only if the answers of the two provers agree on the set v.

The clauses in w'' are "dummy" in some sense. For a fixed w'', the game is simply a copy of the 2P1R game in Raz's verifier. Thus this modified game also has soundness $2^{-\Omega(u)}$. Fix the set w of $(T+1)u$ clauses, hence fixing the question asked to the second prover. The question asked to the first prover can be equivalently viewed as picking a random subset $w' \subseteq w$ of u clauses and taking one variable from each clause in w' giving a set of variables v'. The question to the first prover is $v = v' \cup (w \setminus w')$. Let $\pi^{v,w}$ be the projection that maps an assignment to the set w to its sub-assignment to the set v.

The following lemma gives a crucial property of this construction.

Lemma 4.2.1 (Smoothness property) *For fixed w and any two distinct assignments b_1, b_2 to w,*

$$\Pr_v \left[\pi^{v,w}(b_1) \neq \pi^{v,w}(b_2) \right] \geq 1 - \frac{1}{T}$$

Proof: The assignments b_1, b_2 differ on at least one clause $C_0 \in w$. Note that w' is a random subset of w with $|w'| = u$ and $|w| = (T+1)u$. Hence with probability $1 - \frac{1}{T}$ over the choice of the set w', $C_0 \notin w'$ and consequently $C_0 \in v$. Whenever this happens, the sub-assignments of b_1, b_2 to the set v are distinct. ■

In terms of the Label Cover problem, we can restate Theorem 2.3.2 with the additional smoothness property given by Lemma 4.2.1.

Theorem 4.2.2 *There is an absolute constant $\gamma > 0$ such that for all integer parameters u and T, it is NP-hard to distinguish between the following two cases : A Label Cover problem $\mathcal{L}(G(V,W,E), N, M, \{\pi^{v,w} | (v,w) \in E\})$ with $|M| = 7^{(T+1)u}, |N| = 2^u 7^{Tu}$ has*

- $OPT(\mathcal{L}) = 1$ *OR*

- $OPT(\mathcal{L}) \leq 2^{-\gamma u}$

The Label Cover instance has the following "smoothness property". For every $w \in W$, and $b_1, b_2 \in M$, $b_1 \neq b_2$, if v is a randomly chosen neighbor of w, then

$$\Pr_{v}\left[\ \pi^{v,w}(b_1) \neq \pi^{v,w}(b_2)\ \right] \ \geq\ 1 - \frac{1}{T}$$

It can be assumed that $G(V, W, E)$ is a regular bipartite graph where every vertex in W has degree $\binom{(T+1)u}{u}3^u$ and every vertex in V has degree 5^u.

4.2.2 Definition of Multi-layered Label Cover Problem

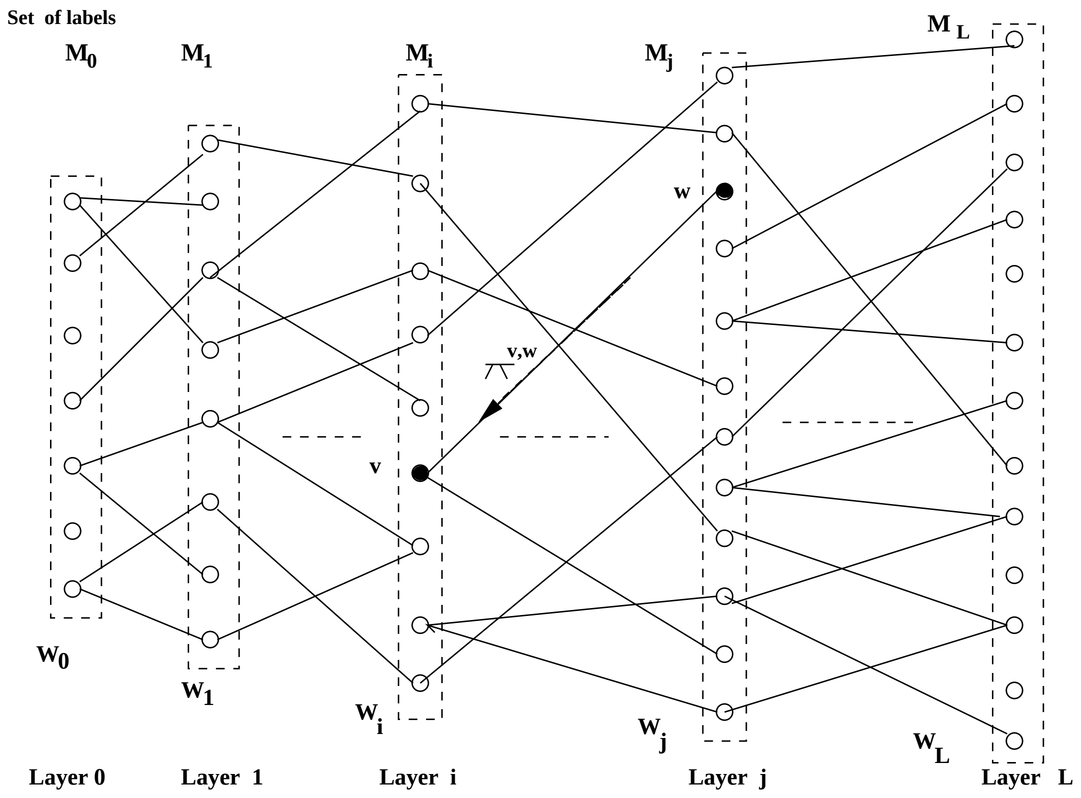


Figure 4.1: Multi-layered Label Cover

Definition 4.2.3 *A multi-layered Label Cover instance (see Fig. 4.1)*

$$\mathcal{L}_{multi}(G, \{W_i\}_{i=0}^{L}, E = \cup_{0 \le i < j \le L} E_{ij}, \{M_i\}_{i=0}^{L}, \{\pi^{v,w}\}_{(v,w) \in E})$$

has the following description : G is a graph whose vertices are partitioned into $L + 1$ layers. The layers are numbered from 0 to L and the set of vertices in layer i is W_i. For $0 \le i < j \le L$, let E_{ij} denote the set of edges between layers i and j. The graph between every pair of layers in a regular bipartite graph. There are no edges between vertices of the same layer.

The goal is to assign labels to vertices of this graph. Vertices in i^{th} layer are supposed to get labels from a set M_i. For $v \in W_i, w \in W_j$ such that $i < j, (v, w) \in E_{ij}$, there is a projection map $\pi^{v,w} : M_j \mapsto M_i$.

An assignment of labels $\Phi : W_i \mapsto M_i$ assigns one label for every vertex in the graph. The assignment is said to satisfy an edge (v, w) if

$$\pi^{v,w}(\Phi(w)) = \Phi(v)$$

Let $OPT(\mathcal{L}_{multi}, i, j)$ denote the maximum fraction of edges satisfied between layers i and j by any labeling.

The following theorem states the construction of the multi-layered smooth Label Cover problem.

Theorem 4.2.4 *There is a poly-time reduction from Gap-3SAT-5 to a multi-layered Label Cover problem*

$$\mathcal{L}_{multi}(G, \{W_i\}_{i=0}^{L}, E = \cup_{0 \le i < j \le L} E_{ij}, \{M_i\}_{i=0}^{L}, \{\pi^{v,w}\}_{(v,w) \in E})$$

and parameters T, u with these properties (think of $T \gg L \gg 1$ and u as independent parameter) :

1. $|M_i| = 2^{(L-i)u} 7^{(T+i)u}$ *for every* $0 \le i \le L$.

2. *For* $i < j < k$ *and* $w_i \in W_i, w_j \in W_j, w_k \in W_k$*, if* $(w_i, w_j) \in E_{ij}, (w_j, w_k) \in E_{jk}$ *then* $(w_i, w_k) \in E_{ik}$*. In fact for any* $w_k \in W_k$*, selecting its random neighbor in layer* i *is same as first selecting its random neighbor* w_j *in layer* j *and then selecting a random neighbor of* w_j *in layer* i*. This property can be generalized to any* t *layers* $i_1 < i_2 < \dots i_t$*. For a vertex* $w_{i_t} \in W_{i_t}$*, selecting its random neighbor in layer* i_1 *is same as successively selecting vertices* $w_{i_{t-1}} \in W_{i_{t-1}}, w_{i_{t-2}} \in W_{i_{t-2}}, \dots w_{i_1} \in W_{i_1}$ *where* $w_{i_{l-1}}$ *is a random neighbor of* w_{i_l} *for* $l = t, t-1, \dots, 2$.

3. *(Completeness) : If Gap-3SAT-5 instance is a YES instance (i.e. satisfiable), then there exists a labeling that satisfies every edge of* $\mathcal{L}_{multi}$.

4. *(Soundness) : If Gap-3SAT-5 instance is a NO instance (i.e. no more than a constant fraction of clauses are satisfiable), then* $OPT(\mathcal{L}_{multi}, i, j) \le 2^{-\gamma(j-i)u} \le 2^{-\gamma u}$*. Here* γ *is an absolute constant same as Theorem 2.3.2.*

5. *(Smoothness Property) :*

   Let $0 \le i < j \le L$ *and* $w \in W_j$*. Let* $b, b' \in M_j$ *be two distinct labels to* w*. If* $v \in W_i$ *is a random neighbor of* w *in layer* i*, then*

$$\Pr{}_v\left[\pi^{v,w}(b) = \pi^{v,w}(b')\right] \le \frac{L}{T}$$

6. *(Weak Expansion Property) :*

Consider any t layers numbered $i_1 < i_2 < \ldots < i_t$ where $t = \lceil 2/\delta \rceil$. Choose any sets $S_{i_l} \subseteq W_{i_l}$ with $|S_{i_l}| \geq \delta |W_{i_l}|$ for $1 \leq l \leq t$. Then there exist two layers numbered i_l and $i_{l'}$ such that the number of edges between the sets S_{i_l} and $S_{i_{l'}}$ is at least a fraction $\delta^2/4$ of the total number of edges between the layers i_l and $i_{l'}$.

4.2.3 The Main Construction

In this section, we prove Theorem 4.2.4. Let a Gap-3SAT-5 instance be given by Theorem 2.1.1. The variables will be denoted by $x_1, x_2, \ldots$ and the clauses by $C_1, C_2, \ldots$. Let T and u be integer parameters.

Defining Layers and Vertices for $\mathcal{L}_{multi}$

For $0 \leq i \leq L$, a type-i vertex corresponds to the union of a set of $(L-i)u$ variables and a set of $(T+i)u$ clauses. Let W_i be the set of of all type-i vertices.

Remark : A type-i vertex is union of a set of variables and a set of clauses. The fact that the components are *sets* and not *tuples* is important. By definition, there is no order associated with elements of a set.

Defining Edges Between Pairs of Layers

For $0 \leq i < j \leq L$, let $v \in W_i$ be a neighbor of $w \in W_j$ if one can obtain v by replacing $(j-i)u$ clauses $\{C_l \mid l = 1, 2, .., (j-i)u\}$ in w by $(j-i)u$ variables $\{x_l \mid l = 1, 2, ..., (j-i)u\}$ such that the variable x_l is contained in the clause C_l for $1 \leq l \leq (j-i)u$.

For $w \in W_j$, a random neighbor of w in layer i is obtained by choosing $(j-i)u$ clauses at random from the $(T+j)u$ clauses in w and replacing each clause by one of the variables appearing in that clause picked at random.

Defining Sets of Labels

A vertex $v \in W_i$ contains $(L-i)u$ variables and $(T+i)u$ clauses and thus a total of $(L-i)u + 3(T+i)u$ variables. A label to vertex v is an assignment to these $(L-i)u + 3(T+i)u$ variables such that the assignment satisfies all the clauses in v. Let M_i denote the set of all satisfying assignments to $v \in W_i$ with $|M_i| = 2^{(L-i)u} 7^{(T+i)u}$.

If v is a type-i neighbor of a type-j vertex w, then every satisfying assignment to w can be restricted to a satisfying assignment to v. Let the map $\pi^{v,w} : M_j \mapsto M_i$ denote this operation of taking a sub-assignment/restriction.

Regularity, Completeness and Soundness

Since the Gap-3SAT-5 instance is regular, it is clear that the bipartite graph between every pair of layers W_i and W_j is regular. Also, Property (2) of Theorem 4.2.4 follows from the way we define edges between two layers.

Completeness is clear. For soundness, we note that between every pair of layers, we have an instance of the smooth 2P1R game described in Section 4.2.1. As noted there, this instance contains copies of the Raz Verifier and hence the soundness is $2^{-\gamma(j-i)u}$.

Proving the Smoothness Property

Fix $w \in W_j$ and two assignments b, b' to w which differ in at least one bit. The projection $\pi^{v,w}$ preserves the variables in w, but replaces some clauses by variables. If b, b' differ on a variable in w, their projections under $\pi^{v,w}$ are still distinct. Otherwise they differ on some clause, say clause C_0. For a choice of a random neighbor v, one replaces at random $(j-i)u$ clauses out of the $(T+j)u$ clauses in w. With probability $1 - \frac{j-i}{T+j} \geq 1 - \frac{L}{T}$, the clause C_0 is *not* replaced and hence projections of b, b' are distinct.

The following lemma states an immediate consequence of the smoothness property.

Lemma 4.2.5 *Let $0 \leq i < j \leq L$ and $w \in W_j$. Let $\beta \subseteq M_j$ be non-empty and $b \in \beta$. If v is a random neighbor of w of type-i, then with probability $1 - |\beta|L/T$ we have,*

$$\forall\, b' \in \beta,\ b' \neq b, \quad \pi^{v,w}(b) \neq \pi^{v,w}(b')$$

Proof: Just apply the previous lemma to every $b' \in \beta, b' \neq b$ and take a union bound. ■

Proving the Weak Expansion Property

Take any $t = \lceil \frac{2}{\delta} \rceil$ layers $i_1 < \ldots < i_t$ and sets $S_{i_l} \subseteq W_{i_l}$ for $1 \leq l \leq t$ such that $S_{i_l} \geq \delta |W_{i_l}|$. Consider a random walk beginning from a uniformly chosen vertex w_{i_t} in layer W_{i_t} and proceeding to a vertex $w_{i_{t-1}} \in W_{i_{t-1}}$ chosen uniformly among the neighbors of w_{i_t}. The random walk continues in a similar way to a vertex $w_{i_{t-2}} \in W_{i_{t-2}}$ chosen uniformly among the neighbors of $w_{i_{t-1}}$ and so on up to a vertex in W_{i_1}. Denote by E_l the indicator variable of the event that the random walk hits a vertex in S_{i_l} in layer i_l. From the regular structure of the multi-layered graph and Property (2) in Theorem 4.2.4, it follows that for every l, $\Pr[E_l] \geq \delta$. Moreover, using the inclusion-exclusion principle, we get:

$$1 \geq \Pr[\vee_{l=1}^{t} E_l] \geq \sum_l \Pr[E_l] - \sum_{l<l'} \Pr[E_l \wedge E_{l'}]$$
$$\geq \lceil \frac{2}{\delta} \rceil \cdot \delta - \binom{t}{2} \max_{l<l'} \Pr[E_l \wedge E_{i'}] \geq 2 - \binom{t}{2} \max_{l<l'} \Pr[E_l \wedge E_{l'}]$$

which implies

$$\max_{l<l'} \Pr[E_l \wedge E_{l'}] \geq 1/\binom{t}{2} \geq \frac{\delta^2}{4}$$

Fix l and l' such that $\Pr[E_l \wedge E_{l'}] \geq \frac{\delta^2}{4}$. This says that a random walk beginning in layer $i_{l'}$ and ending in layer i_l hits both the sets $S_{i_{l'}}$ and S_{i_l} with probability at least $\delta^2/4$. However, such a random walk is same as picking an edge at random from the set of all edges between layers $i_{l'}$ and i_l. Thus the fraction of edges between the sets $S_{i_{l'}}$ and S_{i_l} is at least a fraction $\delta^2/4$ of the total number of edges between layers $i_{l'}$ and i_l.

4.3 Long Codes over $\mathbf{Z}_k$ and Fourier Analysis

As always, we build a PCP verifier by taking a Label Cover instance and plugging in the Long Code. In this chapter, we need to use long codes over the ring Z_k (integers mod k) instead of binary long codes. These codes are defined as a straightforward generalization of binary long codes.

Long code (over $\mathbf{Z}_k$) over a domain M is indexed by all functions $g \in \mathcal{G}$ where

$$\mathcal{G} := \{g \mid g : M \mapsto \{1, \omega, \omega^2, \ldots, \omega^{k-1}\}\}$$

Here ω is the k^{th} root of unity, i.e. $\omega = e^{2\pi i/k}$. The long code B of $b \in M$ is defined as

$$B(g) := g(b) \quad \forall\, g \in \mathcal{G}$$

Consider the space of all "tables" $B : \mathcal{G} \mapsto$ C . In particular, a long code is one such table. Consider the characters χ_β where $\beta : M \mapsto \mathbf{Z}_k$. There is one such character for every β. The character χ_β is a table defined by

$$\chi_\beta(g) := \prod_{y \in M} g(y)^{\beta(y)}$$

The characters form an orthonormal basis under the following definition of inner product of tables. For tables B_1, B_2, define

$$< B_1, B_2 > := \frac{1}{k^{|M|}} \sum_{g \in \mathcal{G}} B_1(g)\overline{B_2(g)} = E_g\left[B_1(g)\overline{B_2(g)}\right]$$

It follows that any table can be expanded as $B = \sum_\beta \widehat{B}_\beta \chi_\beta$ where $\widehat{B}_\beta$ are the Fourier coefficients with $\sum_\beta |\widehat{B}_\beta|^2 = < B, B >$. When $B : \mathcal{G} \mapsto \{1, \omega, \ldots, \omega^{k-1}\}$, we have $\sum_\beta |\widehat{B}_\beta|^2 = 1$. The Fourier coefficients are given by

$$\widehat{B}_\beta = < B, \chi_\beta > = E_g\left[B(g)\overline{\chi_\beta(g)}\right]$$

In particular when $\beta \equiv 0$, the value of the coefficient $\widehat{B}_0$ is just $E_g[B(g)]$.

4.4 Hardness of 3-Uniform Hypergraph Coloring

In this section, we prove Theorem 4.1.1. We construct a PCP verifier that expects a proof over the domain $\mathbf{Z}_3$. The verifier reads 3 symbols from the proof and accepts if and only if not all 3 symbols are equal. We let the locations in the proof to be the vertices of a hypergraph and the tests of the verifier (reading 3 locations) to be the edges of the hypergraph. This defines a 3-uniform hypergraph and we prove the desired completeness and soundness properties.

We reduce the Gap-3SAT-5 instance to an instance $\mathcal{L}_{multi}$ of the multi-layered smooth Label Cover given by Theorem 4.2.4. We expect the proof to contain for every vertex $w \in W_j$, the long code A_w (over $\mathbf{Z}_3$) of a supposed label to w. So this long code is over the domain M_j (the set of labels for layer j). Let $\omega = e^{2\pi i/3}$ be the cube root of unity. The long code is indexed by all functions $g \in \mathcal{F}_j$ where

$$\mathcal{F}_j := \{g \mid g : M_j \mapsto \{1, \omega, \omega^2\}\}$$

The verifier's action is :

- Pick $0 \le i < j \le L$ at random.
- Pick a random vertex $w \in W_j$ and its random neighbor $v \in W_i$ in layer i. Let $\pi^{v,w} : M_j \mapsto M_i$ be the projection between v and w. Let $A = A_v$ and $B = A_w$ be the supposed long codes of labels of vertices v and w. These long codes are indexed by functions $f \in \mathcal{F}_i$ and $g \in \mathcal{F}_j$ respectively.

- Pick random functions $f \in \mathcal{F}_i, g \in \mathcal{F}_j$. Pick a function $\mu \in \mathcal{F}_j$ by defining for every $x \in M_j$

$$\mu(x) := \begin{cases} \omega \text{ with probability } 1/2 \\ \omega^2 \text{ with probability } 1/2 \end{cases}$$

- Let $h \in \mathcal{F}_j$ be defined as $h := \overline{g} \cdot \overline{f \circ \pi^{v,w}} \cdot \mu$

- Accept if and only if

$$\text{Not-All-Equal}(A(f), B(g), B(h))$$

Completeness

In a correct proof, A will be the long code of some $a \in M_i$ and B will be the long code of some $b \in M_j$, with $\pi^{v,w}(b) = a$. In that case

$$A(f) = f(a),\ B(g) = g(b),$$

$$B(h) = h(b) = \overline{g(b) f(\pi^{v,w}(b))}\mu(b) = \overline{g(b) f(a)}\mu(b)$$

Hence $A(f), B(g), B(h)$ cannot all be equal, since μ takes values only in the set $\{\omega, \omega^2\}$. Thus the test always accepts a correct proof and the hypergraph is 3-colorable.

4.4.1 Soundness of PCP

We will show that the size of any independent set in the hypergraph is at most δ fraction of the size of the whole hypergraph. Note that the hypergraph is a $(L + 1)$-layered hypergraph with hyperedges between every pair of layers. The hypergraph is a weighted hypergraph. The vertices in the same layer have equal weight. The total weight of all vertices in any layer is $\frac{1}{L+1}$. Since there are $L + 1$ layers, the total weight of all vertices in the hypergraph is 1.

Let $\mathcal{I}$ be any set of locations in the proof with weight δ. We will show that there is at least a constant fraction of the tests for which all 3 queries lie in the set $\mathcal{I}$. This means that for any set of vertices with weight δ in the hypergraph, at least a constant fraction of hyperedges are contained in this set. Hence there is no independent set of size δ.

Define tables A_v as follows : For every vertex v,

$$A_v(f) := \begin{cases} 1 \text{ if the location } A_v(f) \in \mathcal{I} \\ 0 \text{ otherwise} \end{cases}$$

Remark : The tables A_v are 0-1 tables whereas the proofs are supposed to be over the alphabet $\{1, \omega, \omega^2\}$. However 0-1 tables still make sense in the Fourier analysis and in fact going to 0-1 tables is a trick we exploit.

Call a vertex v "good" if at least $\delta/2$ fraction of the locations in the table A_v are set to 1. Equivalently, v is good if the Fourier coefficient $\widehat{A}_{v,0}$ of the table A_v satisfies

$$\widehat{A}_{v,0} \ \geq \ \delta/2$$

By an averaging argument, at least $\delta/2$ fraction of the vertices v are good. Again by an averaging argument, in at least $\delta/4$ fraction of the layers, at least $\delta/4$ fraction of the vertices are good. The number of such layers is $\geq \ \delta/4 \cdot L \ \geq \ O(1/\delta)$ provided $L > O(1/\delta^2)$.

By the "weak expansion property" of the multi-layered graph (see Theorem 4.2.4), there exist two layers $i_0 < j_0$ such that at least $\delta/4$ fraction of the vertices in each of the two layers are good and the number of edges between the good vertices is at least $\delta\,' = \Omega(\delta^2)$ fraction of the total number of edges between layers i_0 and j_0. Fix these layers i_0 and j_0 for the rest of the proof.

Consider the process of picking a random vertex w from layer j_0 and its random neighbor v from layer i_0. As noted, with probability δ', both w and v are good. Denoting by $A = A_v$ and $B = A_w$, we have

$$E_{v,w}\left[\widehat{A}_0\widehat{B}_0^2\right] \geq (\delta/2)^3\delta' = \delta'' \text{ (say)} \tag{4.1}$$

Now consider the probability that all three queries $A(f), B(g), B(h)$ lie in the set $\mathcal{I}$. Every proof location is 1 or 0 depending on whether it is in the set $\mathcal{I}$ or not. Hence this probability is the expectation

$$E_{v,w,f,g,\mu}\big[A(f)B(g)B(h)\big]$$

which can be expanded using Fourier expansions of tables A and B as

$$E_{v,w,f,g,\mu}\big[\sum_{\alpha,\beta,\gamma}\widehat{A}_\alpha\widehat{B}_\beta\widehat{B}_\gamma\ \chi_\alpha(f)\ \chi_\beta(g)\ \chi_\gamma(\overline{g}\,\overline{f\circ\pi}\mu)\big]$$

where we denoted $\pi = \pi^{v,w}$ for notational convenience. Note that $\beta, \gamma : M_{j_0} \mapsto \mathbf{Z}_3$ and $\alpha : M_{i_0} \mapsto \mathbf{Z}_3$. The expression can be written as

$$E_{v,w,f,g,\mu}\big[\sum_{\alpha,\beta,\gamma}\widehat{A}_\alpha\widehat{B}_\beta\widehat{B}_\gamma\ \chi_\alpha(f)\ \chi_\beta(g)\ \overline{\chi_\gamma(g)}\ \overline{\chi_\gamma(f\circ\pi)}\ \chi_\gamma(\mu)\big] =$$

$$E_{v,w,f,g,\mu}\big[\sum_{\alpha,\beta,\gamma}\widehat{A}_\alpha\widehat{B}_\beta\widehat{B}_\gamma\;\chi_\alpha(f)\;\chi_{\beta-\gamma}(g)\;\overline{\chi_\gamma(f\circ\pi)}\;\chi_\gamma(\mu)\big]$$

This expectation is zero unless $\beta=\gamma$. Also $\chi_\beta(f\circ\pi)=\chi_{\pi_3(\beta)}(f)$ if we let $\pi_3(\beta):M_{i_0}\mapsto\mathbf{Z}_3$ be a function defined as

$$\text{For }\; y\in\; M_{i_0},\quad \pi_3(\beta)(y):=\sum_{x\in M_{j_0}\;:\;\pi(x)=y}\beta(x)\qquad\text{(sum is over }\mathbf{Z}_3\text{)}$$

Hence the expression simplifies to

$$E_{v,w,f,g,\mu}\big[\sum_{\alpha,\beta}\widehat{A}_\alpha\widehat{B}_\beta^2\;\chi_{\alpha-\pi_3(\beta)}(f)\;\chi_\beta(\mu)\big]$$

Again, the expression is zero unless $\alpha=\pi_3(\beta)$. Defining "cardinality" of β as

$$|\beta|=|\{x\mid x\in M_{j_0},\beta(x)\neq 0\}|$$

we get $E_\mu\big[\chi_\beta(\mu)\big]=(-\frac{1}{2})^{|\beta|}$ (verify !). Thus the expression further simplifies to

$$E_{v,w}\big[\sum_\beta\widehat{A}_{\pi_3(\beta)}\;\widehat{B}_\beta^2\;(-\frac{1}{2})^{|\beta|}\big]\tag{4.2}$$

Lemma 4.4.1 *The terms in (4.2) with $\pi_3(\beta)\neq 0$ can be bounded in magnitude by $2^{-\Omega(u)}$.*

Proof: Using a standard argument, we will show that if the terms with $\pi_3(\beta)\neq 0$ have significant magnitude, then there exists a labeling to the Label Cover vertices in layers i_0 and j_0 (see Definition 4.2.3) which satisfies a significant fraction of the edges between these layers. Using Cauchy-Schwartz inequality,

$$\begin{aligned}
& \left| E_{v,w}\left[\sum_{\beta:\pi_3(\beta)\neq 0} \widehat{A}_{\pi_3(\beta)} \widehat{B}_\beta^2 (-\tfrac{1}{2})^{|\beta|}\right]\right| \\
\leq \quad & E_{v,w}\left[\sqrt{\sum_{\beta:\pi_3(\beta)\neq 0} |\widehat{A}_{\pi_3(\beta)}|^2 |\widehat{B}_\beta|^2 (\tfrac{1}{2})^{2|\beta|}}\sqrt{\sum_\beta |\widehat{B}_\beta|^2}\right] \\
\leq \quad & \sqrt{E_{v,w}\left[\sum_{\beta:\pi_3(\beta)\neq 0} |\widehat{A}_{\pi_3(\beta)}|^2 |\widehat{B}_\beta|^2 \tfrac{1}{|\beta|}\right]} \qquad (4.3)
\end{aligned}$$

Note that A, B are 0-1 tables and thus $\sum_\alpha |\widehat{A}_\alpha|^2 \leq 1$ and $\sum_\beta |\widehat{B}_\beta|^2 \leq 1$. The labeling is defined as follows : For a vertex $w \in W_{j_0}$, pick β with probability $|\widehat{B}_\beta|^2$, pick a random $x \in M_{j_0}$ such that $\beta(x) \neq 0$ and define label$(w) = x$. For a vertex $v \in W_{i_0}$, pick α with probability $|\widehat{A}_\alpha|^2$, pick a random $y \in M_{i_0}$ such that $\alpha(y) \neq 0$ and define label$(v) = y$. The summation in (4.3) is precisely the probability that $\pi($label$(w)) =$ label(v). This is bounded by $OPT(\mathcal{L}_{multi}, i_0, j_0) \leq 2^{-\Omega(u)}$ as claimed (soundness property in Theorem 4.2.4). ∎

Next, we note that the terms in (4.2) with $|\beta| \geq \log T$ have magnitude at most $(\frac{1}{2})^{\log T} = \frac{1}{T}$ which can be assumed to be negligible by choosing T large enough.

Thus we are left with terms where $\pi_3(\beta) = 0$ and $|\beta| \leq \log T$. Fix w and consider the case when $\beta \neq 0$, i.e. there exists $x \in M_{j_0}$ such that $\beta(x) \neq 0$. Lemma 4.2.5 shows that over the choice of a random neighbor v, with probability $1 - \log T \cdot L/T$, we have

$$\forall\, x' \text{ s.t. } \beta(x') \neq 0,\ x \neq x' \ \text{ we have } \ \pi(x) \neq \pi(x')$$

Whenever this happens, $\pi_3(\beta) \neq 0$. Hence the probability that $\pi_3(\beta) = 0$ is at most $\frac{\log T \cdot L}{T}$ which is negligible. Thus the terms with $\pi_3(\beta) = 0, \beta \neq 0$ and $|\beta| \leq \log T$ have negligible magnitude. Finally we are left with the single term

$$E_{v,w}\left[\widehat{A}_0\widehat{B}_0^2\right]$$

which is at least δ'' as observed before (Equation (4.1)). Assuming the magnitude of all the terms neglected to be at most $\delta''/2$, it follows that at least $\delta''/2$ fraction of the tests between layers i_0 and j_0 have all 3 queries in the set $\mathcal{I}$. In particular, $\mathcal{I}$ cannot be an independent set. This completes the proof.

Theorem 4.1.2 is proved by plugging in appropriate super-constant values of the parameters L, T, u in the analysis. We need $L = O(1/\delta^2), T = O(\log(1/\delta)/\delta^7), u = O(\log(1/\delta))$. The size of the hypergraph produced is roughly $N = n^{Tu}3^{7^{Tu}}$. Choosing $\delta = (\log\log n)^{-1/8}$ we have $N = n^{O(\log\log n)}$ and it is either 3-colorable or has no independent set of size $\delta \approx (\log\log N)^{-1/8}$.

4.5 Hardness of the Problem NAE$_{3,k}$

In this section, we prove Theorem 4.1.3. It suffices to construct a PCP verifier that reads 3 symbols from a proof over the alphabet $\mathbf{Z}_k$, accepts if not all 3 symbols are equal, has perfect completeness and soundness $\frac{k^2-1}{k^2} + \epsilon$ where $\epsilon > 0$ is an arbitrary constant.

The verifier is based on the construction of the multi-layered Label Cover problem given by Theorem 4.2.4. However we do not need the “weak expansion property” of this construction. We make a crucial use of Lemma 4.2.5.

After picking a vertex $w_j \in W_j$ and its neighbor $w_i \in W_i$ with a suitable distribution, the verifier’s test is very similar to the test given in Section 4.4. We use long codes over $\mathbf{Z}_k$. Let $\omega = e^{2\pi i/k}$ be the complex k^{th} root of unity.

Given an instance $\mathcal{L}_{multi}$, the action of the verifier is

- Pick a random vertex $w_L \in W_L$ from the layer L. For $i = L, L-1, L-2, \ldots, 1$, let $w_{i-1} \in W_{i-1}$ be a random neighbor of the (already chosen) vertex $w_i \in W_i$.
- Let $A_i = A_{w_i}$ be the supposed long code of a supposed label of the vertex w_i. Let M_i be the set of labels for vertices in layer W_i. For $0 \leq i < j \leq L$, let $\pi^{i,j} = \pi^{w_i, w_j} : M_j \mapsto M_i$ be the projection function between w_i and w_j.
- Pick $0 \leq i < j \leq L$ at random.
- Pick functions $f : M_i \mapsto \{1, \omega, \omega^2, \ldots, \omega^{k-1}\},\ g : M_j \mapsto \{1, \omega, \omega^2, \ldots, \omega^{k-1}\}$ at random.
- Pick a function $\mu : M_j \mapsto \{\omega, \omega^2\}$ by defining for every $x \in M_j$

$$\mu(x) := \begin{cases} \omega \text{ with probability } 1/2 \\ \omega^2 \text{ with probability } 1/2 \end{cases}$$

- Let $h : M_j \mapsto \{1, \omega, \omega^2, \ldots, \omega^{k-1}\}$ be defined as $h := \overline{g} \cdot (f \circ \pi^{i,j})^2 \cdot \mu$
- Accept if and only if

$$\text{Not-all-equal}(A_i(f), A_j(g), A_j(h))$$

Completeness

It is easy to see that the test always accepts a correct proof constructed by taking a correct labeling to the multi-layered Label Cover instance $\mathcal{L}_{multi}$ and using correct long codes.

In a correct proof, A_i, A_j are long codes of some labels $a \in M_i, b \in M_j$ with $\pi^{i,j}(b) = a$. Therefore

$$A_i(f) = f(a), A_j(g) = g(b), A_j(h) = h(b) = \overline{g(b)}(f(\pi^{i,j}(b)))^2\mu(b) = \overline{g(b)}f(a)^2\mu(b)$$

The three symbols read by the verifier cannot all be equal, since $\mu(b) = \omega$ or ω^2.

4.5.1 Soundness of PCP

We will show that if the Gap-3SAT-5 instance is a NO instance, the test accepts with probability at most $\frac{k^2-1}{k^2} + \epsilon$. The following lemma can be easily checked.

Lemma 4.5.1 *If $x, y, z \in \{1, \omega, \omega^2, \ldots, \omega^{k-1}\}$, then the expression*

$$1 - \frac{1}{k^2} \sum_{r_1+r_2+r_3=0} x^{r_1} y^{r_2} z^{r_3} \qquad (r_1, r_2, r_3 \in \mathbf{Z}_k)$$

is 0 if $x = y = z$ and 1 otherwise.

From this lemma, it is clear that the acceptance probability of the verifier is

$$1 - \frac{1}{k^2} \sum_{r_1+r_2+r_3=0} E_{\substack{w_L, w_{L-1}, \ldots, w_0, \\ i,j,f,g,\mu}} \left[A_i(f)^{r_1} A_j(g)^{r_2} A_j(h)^{r_3}\right]$$

Fix $w_L, w_{L-1}, \ldots, w_0, i, j$ for the time being and consider the expectation over f, g, μ. For notational convenience, we will drop the $E[]$ notation. Expanding the tables $A_i^{r_1}, A_j^{r_2}, A_j^{r_3}$ by their Fourier expansions, we get

$$1 - \frac{1}{k^2} \sum_{r_1+r_2+r_3=0} \sum_{\alpha,\gamma,\beta} \widehat{A}^{r_1}_{i,\alpha} \widehat{A}^{r_2}_{j,\gamma} \widehat{A}^{r_3}_{j,\beta} \quad \cdot \chi_\alpha(f)\chi_\gamma(g)\chi_\beta(\overline{g}(f \circ \pi^{i,j})^2\mu)$$

The expectation over g is zero unless $\beta = \gamma$. Let $\pi = \pi^{i,j}$ and let $\pi_k(\beta)$ be a function $\pi_k(\beta) : M_i \mapsto \mathbf{Z}_k$ defined as

$$\forall\, y \in M_i,\;\; \pi_k(\beta)(y) := \sum_{x \in M_j : \pi(x) = y} \beta(x) \qquad (\text{sum is over } \mathbf{Z}_k)$$

With this definition, $\chi_\beta((f \circ \pi)^2) = \chi_{2\beta}(f \circ \pi) = \chi_{2\pi_k(\beta)}(f)$ and hence the expectation over f is zero unless $\alpha + 2\pi_k(\beta) = 0$. Thus the acceptance probability is

$$1 - \frac{1}{k^2} \sum_{r_1 + r_2 + r_3 = 0} \sum_{\beta} \widehat{A}^{r_1}_{i,-2\pi_k(\beta)} \widehat{A}^{r_2}_{j,\beta} \widehat{A}^{r_3}_{j,\beta} \cdot E_\mu[\chi_\beta(\mu)] \tag{4.4}$$

Lemma 4.5.2 *Let $\beta : M \mapsto \mathbf{Z}_k$ and pick a function $\mu : M \mapsto \{\omega, \omega^2\}$ by defining $\mu(x)$ to be a uniformly chosen value among the two possible values for every $x \in M$. Let $|\beta| := |\{x \in M \mid \beta(x) \neq 0\}|$. Then*

- *If $\beta = 0$, $E_\mu[\chi_\beta(\mu)] = 1$.*
- $|E_\mu[\chi_\beta(\mu)]| \leq (1 - \Omega(\frac{1}{k^2}))^{|\beta|}$
- *If k is even and if there exists $x_0 \in M$ such that $\beta(x_0) = k/2$, then $E_\mu[\chi_\beta(\mu)] = 0$.*

Proof: We note that the given expectation is

$$E_\mu[\prod_{x \in M} \mu(x)^{\beta(x)}] = \prod_{x \in M} E_\mu[\mu(x)^{\beta(x)}]$$

The first claim is obvious. For the second claim, we show that for every x such that $\beta(x) \neq 0$, the inner expectation has absolute value $\leq 1 - \Omega(\frac{1}{k^2})$. Let $r = \beta(x)$. Because of the way $\mu()$ is defined, the inner expectation is $\frac{1}{2}(\omega^r + \omega^{2r})$ and the claim follows. Note also that if k is even and $r = k/2$, then this expectation vanishes. This proves the third claim. ■

Now we analyze different terms in expression (4.4). Similar to Lemma 4.4.1, the terms with $-2\pi_k(\beta) \neq 0$ can be used to define a "good" labeling to layers i and j of Label Cover instance. Therefore these terms can be assumed to be arbitrarily small in magnitude. The terms with $|\beta| \geq O(k^2 \log T)$ have magnitude at most $\frac{1}{T}$ which is negligible.

Now consider the terms with $\beta \neq 0, -2\pi_k(\beta) = 0$ and $|\beta| \leq O(k^2 \log T)$. Using Lemma 4.2.5, for a fixed w_j, over the choice of a random neighbor w_i, the probability that $\pi_k(\beta) = 0$ is at most $\frac{O(k^2 \log T)\cdot L}{T}$ which is negligible. Hence these terms can also be ignored. One intricate detail here is that when k is even, thanks to third claim in Lemma 4.5.2, we can ignore βs such that $\exists\ x \in M,\ \beta(x) = k/2$. Such βs could have been troublesome : even if $\beta \neq 0$, it could happen that $2\beta = 0$.

Thus we are left with only those terms where $\beta = 0$. The acceptance probability is

$$\Pr[\text{acc}] = 1 - \frac{1}{k^2} \sum_{r_1+r_2+r_3=0} \widehat{A}_{i,0}^{r_1} \widehat{A}_{j,0}^{r_2} \widehat{A}_{j,0}^{r_3} + \epsilon' \tag{4.5}$$

where ϵ' takes into account the terms neglected. Let

$$\widehat{A}_{i,0} = \sum_{l=0}^{k-1} a_{il} \cdot \omega^l \tag{4.6}$$

where a_{il} are non-negative real numbers with $\sum_{l=0}^{k-1} a_{il} = 1$. Note that a_{il} is just the fraction of entries in the table A_i which are equal to ω^l. It follows that

$$\widehat{A}_{i,0}^{r} = \sum_{l=0}^{k-1} a_{il} \cdot \omega^{rl}$$

Hence the summation in (4.5) can be written as

$$\sum_{r_1+r_2+r_3=0} \sum_{l_1,l_2,l_3} a_{il_1} a_{jl_2} a_{jl_3} \omega^{r_1l_1+r_2l_2+r_3l_3} = \sum_{l_1,l_2,l_3} a_{il_1} a_{jl_2} a_{jl_3} \sum_{r_1+r_2+r_3=0} \omega^{r_1l_1+r_2l_2+r_3l_3}$$

The inner summation is 0 unless $l_1 = l_2 = l_3$ and k^2 otherwise. So the sum reduces to

$$k^2 \sum_{l=0}^{k-1} a_{il} a_{jl}^2$$

Recall that the verifier first picks vertices $W_L, W_{L-1}, \ldots, W_0$ and then picks i, j at random. Taking expectation over the choice of $0 \le i, j \le L$, applying Lemma 4.5.3, and choosing L large enough, we get

$$\Pr[\text{acc}] = 1 - E_{\,0 \le i < j \le L}\big[\sum_{l=0}^{k-1} a_{il} a_{jl}^2\big] + \epsilon'$$

$$\le 1 - \frac{1}{k^2} + \frac{2}{L} + \epsilon' \le 1 - \frac{1}{k^2} + 2\epsilon'$$

Lemma 4.5.3 *Let* $\{a_{il} : 0 \le i \le L;\ 0 \le l \le k-1\}$ *be a collection of non-negative reals such that*

$$\sum_{l=0}^{k-1} a_{il} = 1 \quad \forall\, i$$

Then we have

$$E_{\,0 \le i < j \le L}\big[\sum_{l=0}^{k-1} a_{il}\ a_{jl}^2\big] \ge \frac{1}{k^2} - \frac{2}{L}$$

Proof: Let $a_{il} = (1 + b_{il})/k$ so that $\sum_{l=0}^{k-1} b_{il} = 0$ for every i. Also $-1 \le b_{il} \le k - 1$.

$$a_{il} a_{jl}^2 = \frac{1}{k^2} a_{il} (1 + b_{jl})^2 = \frac{1}{k^2} a_{il} (1 + 2b_{jl} + b_{jl}^2) \ge \frac{1}{k^2} a_{il} (1 + 2b_{jl})$$

$$= \frac{1}{k^3} (1 + b_{il})(1 + 2b_{jl}) = \frac{1}{k^3} (1 + b_{il} + 2b_{jl} + 2b_{il} b_{jl})$$

Summing over l and noting that $\sum_{l=0}^{k-1} b_{il} = 0$, we get

$$\sum_{l=0}^{k-1} a_{il} a_{jl}^2 \geq \frac{1}{k^2} + \frac{2}{k^3} \sum_{l=0}^{k-1} b_{il} b_{jl}$$

Finally, $$E_{\,0 \leq i < j \leq L}\big[b_{il} b_{jl}\big] = \frac{1}{2\binom{L+1}{2}} \big((\sum_{i=0}^{L} b_{il})^2 - \sum_{i=0}^{L} b_{il}^2 \big)$$

$$\geq \frac{1}{2\binom{L+1}{2}} (-(L+1)(k-1)^2) \geq -\frac{k^2}{L}$$

■

Remark

Lemma 4.5.3 has a very nice interpretation in terms of a randomized scheme of coloring. Note that the hypergraph we construct has $L+1$ layers and there are edges between every pair of layers i and j. Each edge has one vertex in layer i and two vertices in layer j. Consider the following scheme of coloring the vertices of the hypergraph with k colors. Fix non-negative real numbers $\{a_{il} \mid 0 \leq i \leq L,\ 0 \leq l \leq k-1\}$ such that for every i, $\sum_{l=0}^{k-1} a_{il} = 1$. Color every vertex in layer i with color l with probability a_{il}. Clearly, the fraction of edges that are mono-chromatic is precisely $E_{\,0 \leq i < j \leq L}\big[\sum_{l=0}^{k-1} a_{il}\, a_{jl}^2\big]$.

Lemma 4.5.3 claims that *any* such randomized scheme of coloring must leave at least a fraction $\frac{1}{k^2} - o(1)$ edges mono-chromatic. Now such a claim has to be true since we want to prove that *any* coloring of the hypergraph must have at least $\frac{1}{k^2} - \epsilon$ fraction of the edges mono-chromatic ! Equation (4.6) says that the coefficients $\widehat{A}_{i,0}$ correspond to a randomized scheme of coloring. Thus the Fourier coefficients do have a meaningful interpretation and not just a part of crazy algebraic calculation !

Chapter 5

Hardness of Coloring 4-Uniform Hypergraphs

There is a huge gap between the known algorithmic and hardness results for (Hyper)Graph Coloring. All known algorithms for coloring k-colorable (hyper)graphs require $n^{\Omega(1)}$ colors where n is the number of vertices. On the hardness side, in Chapter 3, we obtained a modest lower bound of $k^{\Omega(\log k)}$ colors for graph coloring. Once we move to hypergraphs, we have reasonable lower bounds : Guruswami et al's $\Omega(\frac{\log \log n}{\log \log \log n})$ bound (see [55]) for coloring 4-uniform hypergraphs and our $(\log \log n)^{\Omega(1)}$ bound for coloring 3-uniform hypergraphs in Chapter 4. Still, these lower bounds fall way short of the polynomial upper bound.

In this chapter, we prove a much stronger result. We show hardness of coloring k-colorable 4-uniform hypergraphs with $(\log n)^{\Omega(k)}$ colors. The function $(\log n)^{\Omega(k)}$ exceeds every polylog function as k increases and gives evidence that the right answer for (hyper)graph coloring problem might be super-polylogarithmic.

We introduce a new code called Split Code. This is a length efficient variant of long code and yields proofs of much smaller size. The code is designed to exploit special structure of Label Cover instances obtained by parallel repetition. We use Split Codes over non-boolean domain and make a novel use of this feature.

5.1 Results and Techniques

The main result in this chapter is :

Theorem 5.1.1 *There exists an absolute constant $c > 0$ such that for every fixed integer $p \geq 5$, it is hard to distinguish whether an n-vertex 4-uniform hypergraph is p-colorable or it contains no independent set of (relative) size $(\log n)^{-cp}$ unless $\mathrm{NP} \subseteq \mathrm{DTIME}(2^{(\log n)^{O(1)}})$. In particular, it is hard to color p-colorable 4-uniform hypergraphs with $(\log n)^{cp}$ colors unless $\mathrm{NP} \subseteq \mathrm{DTIME}(2^{(\log n)^{O(1)}})$.*

Techniques

We prove Theorem 5.1.1 by constructing a PCP verifier that reads 4 symbols from a proof over alphabet $GF(p)$ and accepts if and only if not all 4 symbols are equal. The hypergraph is constructed by taking the positions in the proof as vertices and the tests of the verifier as hyperedges.

The main ingredient in the PCP construction is a new code which we call the Split Code. This is a variation of the long code, but much shorter in length and reduces the proof size significantly. Split Codes enable us to exploit the special structure of the Label Cover problem constructed via Raz's Parallel Repetition Theorem. Recall that Label Cover problem (Definition 2.3.1) consists of a bipartite graph and asks for a labeling to its vertices. The labeling is required to satisfy certain constraints given by maps $\pi : M \mapsto N$

where M, N are sets of labels for the two sides of the bipartite graph. The instance of Label Cover obtained by Parallel Repetition has a "product structure". For a parameter t, we can assume that $M = M_1 \times M_2 \times \ldots M_t$, $N = N_1 \times N_2 \times \ldots N_t$ and π is given by componentwise projections $\pi_i : M_i \mapsto N_i$ for $1 \leq i \leq t$.

Using the Split Codes in our construction requires us to employ Split Codes over large domain i.e. $GF(p)$. We make a novel and essential use of the fact that we are working over non-boolean domain. The analysis in this chapter is quite simple and it demonstrates how things become easier once we move to 4-uniform hypergraphs instead of graphs or 3-uniform hypergraphs.

The Split Codes can be seen as a general technique for reducing proof size. Håstad and Srinivasan [63] use Split Codes to show the following result for Max-3-Lin-2. It is strengthening of Theorem 2.0.2.

Theorem 5.1.2 [63] *There exists a constant $\gamma > 0$ such that it is hard to distinguish between instances of Max-3-Lin-2 where there exists an assignment that satisfies $1 - \frac{1}{(\log n)^\gamma}$ fraction of equations or no assignment can satisfy more than $\frac{1}{2} + \frac{1}{2^{(\log n)^\gamma}}$ fraction of equations unless $\mathrm{NP} \subseteq \mathrm{DTIME}(2^{(\log n)^{O(1)}})$.*

Guruswami et al [55] give a reduction from 4-uniform hypergraphs to q-uniform hypergraphs for any $q \geq 5$. One can use this reduction and extend Theorem 5.1.1 to :

Theorem 5.1.3 *There exists an absolute constant $c > 0$ such that for fixed integers $p \geq 5$ and $q \geq 4$, it is hard to color p-colorable q-uniform hypergraphs with $(\log n)^{cp}$ colors unless $\mathrm{NP} \subseteq \mathrm{DTIME}(2^{(\log n)^{O(1)}})$.*

5.2 Preliminaries

As mentioned before, we exploit the "product structure" of the Label Cover problem given by parallel repetition.

5.2.1 Split Label Cover Problem

Definition 5.2.1 *Split Label Cover $\mathcal{L}_{split}(G(V,W,E), N, M, \{\pi^{v,w}\}, t)$ is the following problem : We are given a bipartite graph $G = (V,W,E)$ such that all vertices in V have the same degree and all vertices in W have the same degree. There is a set of labels $M = M_1 \times M_2 \ldots \times M_t$ for vertices in W and a set of labels $N = N_1 \times N_2 \ldots \times N_t$ for vertices in V. For every edge $(v,w) \in E$, there is a projection function $\pi^{v,w} : M \mapsto N$ which is given by componentwise projections $\pi^{v,w} = (\pi^{v,w,1}, \ldots, \pi^{v,w,t})$ where $\pi^{v,w,i} : M_i \mapsto N_i$ and*

$$\pi^{v,w}(b_1, b_2, \ldots, b_t) = (\pi^{v,w,1}(b_1), \ldots, \pi^{v,w,t}(b_t)) \qquad \forall\, (b_1, b_2, \ldots, b_t) \in M$$

The goal is to assign a labeling $\Phi : V \mapsto N$ and $\Phi : W \mapsto M$ such that

$$\forall\, (v,w) \in E, \quad \pi^{v,w}(\Phi(w)) = \Phi(v)$$

We say that an edge is "satisfied" if this condition holds for that edge. The optimal value $OPT(\mathcal{L}_{split})$ of the Label Cover problem is the maximum fraction of edges that can be satisfied by any labeling.

The following theorem is just a restatement of Theorem 2.3.2, with the extra observation that ut parallel repetitions can be thought of as t-wise product of u-parallel repetitions each.

Theorem 5.2.2 *There exists an absolute constant $\gamma > 0$ such that for all integers u, t, there exists a reduction from Gap-3SAT-5 formula ϕ of size n to a Split Label Cover instance $\mathcal{L}_{split}(G(V, W, E), N, M, \{\pi^{v,w}\}, t)$ such that*

1. $|V|, |W| \leq n^{O(ut)}$
2. $|M_i| = 7^u, |N_i| = 2^u \;\forall\; 1 \leq i \leq t$.
3. *If ϕ is a YES instance, $OPT(\mathcal{L}_{split}) = 1$.*
4. *If ϕ is a NO instance, $OPT(\mathcal{L}_{split}) \leq 2^{-\gamma ut}$.*

5.2.2 The Split Code

Similar to the standard paradigm developed in earlier chapters, our PCP verifier reduces a Gap-3SAT-5 instance to the Split Label Cover instance $\mathcal{L}_{split}$ given by Theorem 5.2.2 and expects the proof to contain encodings of labels of all vertices in V and W. The previous PCP constructions use the long code whereas we use the Split Code which we define next.

Let ω be the basic p^{th} root of unity, i.e. $\omega = e^{2\pi i/p}$. In this chapter, the variables β, g, b stand for tuples and variables β_i, g_i, b_i stand for the i^{th} components of these tuples respectively.

Definition 5.2.3 *The Split Code (over $GF(p)$) on a domain*

$$M = M_1 \times M_2 \ldots \times M_t$$

is indexed by all tuples $g = (g_1, g_2, \dots, g_t)$ *where* $g_i : M_i \mapsto \{1, \omega, \omega^2, \dots, \omega^{p-1}\}$*. The Split Code* B *of an element* $b \in M,\ b = (b_1, b_2, \dots, b_t),\ b_i \in M_i,$ *is defined as*

$$B(g) := \prod_{i=1}^{t} g_i(b_i)$$

The set of all functions $g_i : M_i \mapsto \{1, \omega, \dots, \omega^{p-1}\}$ *is denoted by* $\mathcal{G}_i$*.*

Note that the length of the Split Code on M is $p^{|M_1|+|M_2|+\ldots+|M_t|}$. On the other hand, the length of the long code on M will be much larger i.e. $p^{|M|} = p^{|M_1|\cdot|M_2|\cdot\ldots\cdot|M_t|}$.

5.2.3 Fourier Analysis of Split Codes

Let $\mathcal{G}_1, \dots, \mathcal{G}_t$ be as in Definition 5.2.3. Consider the complex vector space of all functions

$$B : \mathcal{G} = \mathcal{G}_1 \times \mathcal{G}_2 \times \ldots \times \mathcal{G}_t \mapsto \mathbf{C}$$

where the addition of two functions is defined as pointwise addition. This is a complex vector space of dimension $|\mathcal{G}| = p^{|M_1|+|M_2|+\ldots+|M_t|}$. Define an inner product on this space as

$$< B_1, B_2 > = \frac{1}{|\mathcal{G}|} \sum_{g \in \mathcal{G}} B_1(g)\overline{B_2(g)} = E_g[B_1(g)\overline{B_2(g)}]$$

We will identify an orthonormal basis for this vector space. Let β_i be a function

$$\beta_i : M_i \mapsto GF(p)$$

and $\beta = (\beta_1, \beta_2, \dots, \beta_t)$ be a tuple. The character χ_β is a function $\chi_\beta : \mathcal{G} \mapsto \{1, \omega, \dots, \omega^{p-1}\}$ defined as

$$\text{For } g = (g_1, \dots, g_t), \quad \chi_\beta(g) = \prod_{i=1}^{t} \prod_{x_i \in M_i} g_i(x_i)^{\beta_i(x_i)}$$

Lemma 5.2.4 *The characters χ_β form an orthonormal basis.*

Proof: We clearly have $< \chi_\beta, \chi_\beta > = 1$. For $\beta = (\beta_1, \ldots, \beta_t),\ \gamma = (\gamma_1, \ldots, \gamma_t)$ and $\beta \neq \gamma$, there exists an index i_0 and an element $b_{i_0} \in M_{i_0}$ such that $\beta_{i_0}(b_{i_0}) \neq \gamma_{i_0}(b_{i_0})$. Hence

$$\begin{aligned} < \chi_\beta, \chi_\gamma > &= E_{g_1,\ldots,g_t}\left[\prod_{i=1}^{t}\prod_{x_i \in M_i} g_i(x_i)^{\beta_i(x_i)-\gamma_i(x_i)}\right] \\ &= \prod_{i=1}^{t}\prod_{x_i \in M_i} E_{g_i}\left[g_i(x_i)^{\beta_i(x_i)-\gamma_i(x_i)}\right] \end{aligned}$$

The inner expectation is zero when $i = i_0$ and $x_i = b_{i_0}$. ■

Hence every function $B : \mathcal{G} \mapsto \{1, \omega, \ldots, \omega^{p-1}\}$ can be written as

$$B = \sum_\beta \widehat{B}_\beta \chi_\beta \text{ with Parseval's identity } \sum_\beta |\widehat{B}_\beta|^2 = 1$$

5.2.4 Equality Folding of Split Codes

Recall that the proof for the PCP verifier is supposed to contain Split Codes of labels of all vertices in V and W in the Label Cover instance.

It turns out that we can identify pairs of indices (g, g') such that for every correct Split Code B on a set M, we have $B(g) = B(g')$. So we can identify the indices g and g' together which we call the "equality folding". The positions in the Split Code are going to be vertices of a hypergraph and this identification corresponds to merging the vertices for positions g and g' into one vertex.

Definition 5.2.5 *For a function $\beta_i : M_i \mapsto GF(p)$, define*

$$weight(\beta_i) = \sum_{x \in M_i} \beta_i(x) \quad (in\; GF(p))$$

For a tuple $\beta = (\beta_1, \ldots, \beta_t)$, let $\;weight(\beta) = \sum_{i=1}^{t} weight(\beta_i)$.

Definition 5.2.6 *We call a function $B : \mathcal{G} \mapsto \{1, \omega, \ldots, \omega^{p-1}\}$ equality folded if for $e_1, \ldots, e_t \in GF(p)$*

$$e_1 + e_2 + \ldots + e_t = 0 \implies B(g_1, \ldots, g_t) = B(\omega^{e_1} g_1, \ldots, \omega^{e_t} g_t)$$

Note that a correct Split Code is always equality folded. The notion of equality folding forces only equality constraints and it is alright for coloring results. There are other notions of folding (see for instance Definition 2.4.3) which cannot be used in coloring results, and in fact this turns out to be the biggest difficulty in proving hardness of hypergraph coloring. The following is a crucial consequence of equality folding :

Lemma 5.2.7 *If a function $B : \mathcal{G} \mapsto \{1, \omega, \ldots, \omega^{p-1}\}$ is equality folded then $\;\forall\, \beta = (\beta_1, \ldots, \beta_t)$,*

$$\widehat{B}_\beta \neq 0 \implies weight(\beta_1) = weight(\beta_2) = \ldots = weight(\beta_t)$$

Proof: Let $e = (e_1, \ldots, e_t)$ be a vector chosen at random such that $e_1 + e_2 + \ldots + e_t = 0$. There are p^{t-1} choices for e. We have

$$\begin{aligned}
\widehat{B}_\beta = & \quad E_{g=(g_1,\ldots,g_t)}\left[B(g_1,\ldots,g_t)\overline{\chi_\beta(g_1,\ldots,g_t)}\right] \\
= & \frac{1}{p^{t-1}}E_g\left[\sum_e B(\omega^{e_1}g_1,\ldots,\omega^{e_t}g_t)\overline{\chi_\beta(\omega^{e_1}g_1,\ldots,\omega^{e_t}g_t)}\right] \\
= & \quad \frac{1}{p^{t-1}}E_g\left[\sum_e B(g)\overline{\chi_\beta(g)}\,\omega^{-\sum_{i=1}^t e_i \cdot weight(\beta_i)}\right]
\end{aligned}$$

As e ranges over all vectors with $e_1 + e_2 + \ldots + e_t = 0$, the inner sum is zero unless $weight(\beta_1) = weight(\beta_2) = \ldots = weight(\beta_t)$. ■

5.3 The PCP Construction

Now we are ready to define the PCP test that proves Theorem 5.1.1 (see Fig. 5.1). The verifier reads 4 symbols over alphabet $GF(p)$ and accepts if and only if not all 4 symbols are equal. The hypergraph is constructed by letting the positions in the proof to be vertices and tests of the verifier as edges. The PCP has perfect completeness and therefore (in completeness case) the hypergraph is p-colorable. In soundness case, we show that there is no large independent set.

Let $t = \lfloor \frac{p-1}{3} \rfloor$ and $\mathcal{L}_{split}(G(V,W,E), N, M, \{\pi^{v,w}\}, t)$ be the Split Label Cover instance given by Theorem 5.2.2. The verifier expects the proof to contain Split Codes over $GF(p)$ of the labels of all vertices $w \in W$.

Remark : The verifier does not need encodings of labels of the vertices in V. In earlier chapters, we had tests that checked consistency between long codes for vertices v, w where $v \in V, w \in W$. However, here we will check consistency between long codes for $w, w' \in W$ which have a common neighbor $v \in V$. We need to check that the labels of

w, w' project to the same label for v via the respective projection maps $\pi^{v,w}$ and $\pi^{v,w'}$. This technique was introduced by Håstad [60] and also used in [55].

It is easy to motivate this technique. Let us say we instead had tests checking consistency between vertices in V and W. Then the hypergraph would split into two layers with every hyperedge containing vertices from both the layers. Such a hypergraph is clearly 2-colorable and the construction would fail.

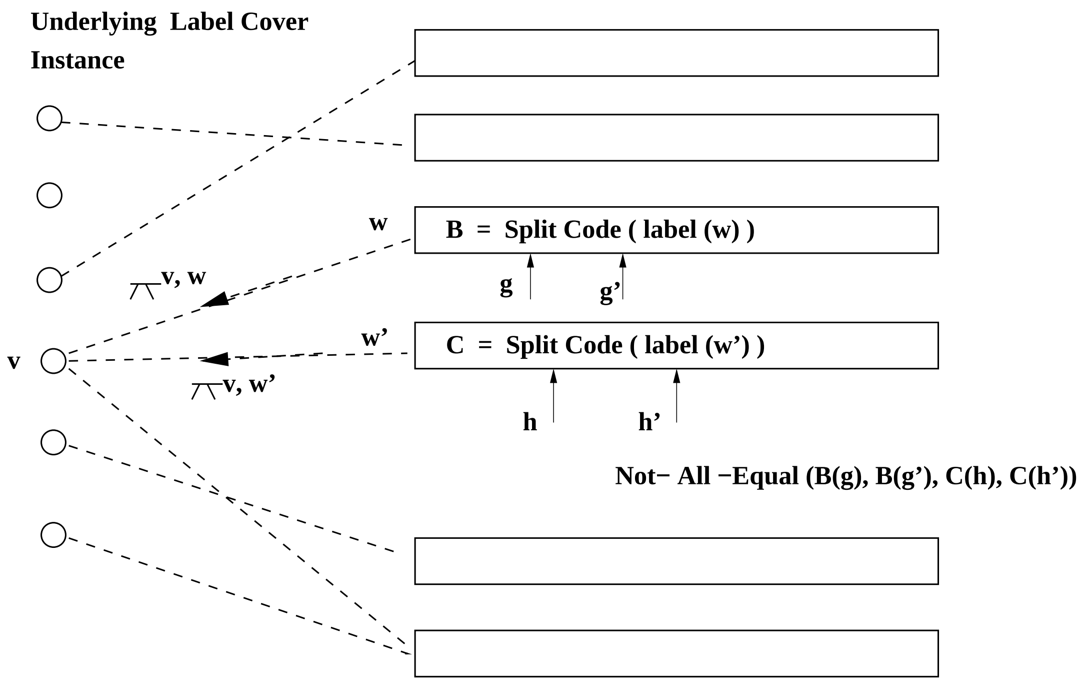

Underlying Label Cover Instance
w
B = Split Code (label (w))
$\pi^{v,w}$
g
g'
w'
C = Split Code (label (w'))
v
$\pi^{v,w'}$
h
h'
Not− All −Equal (B(g), B(g'), C(h), C(h'))

Figure 5.1: PCP for 4-Uniform Hypergraph Coloring

The verifier proceeds as follows :

1. Pick a vertex $v \in V$ at random and two of its neighbors $w, w' \in W$ at random. Let B be the supposed Split Code of the supposed label of w and C be the supposed Split Code of the supposed label of w'.

2. Pick random functions $g_i, h_i : M_i \mapsto \{1, \omega, \ldots, \omega^{p-1}\}$ and $f_i : N_i \mapsto \{1, \omega, \ldots, \omega^{p-1}\}$ for $1 \leq i \leq t$. Let $g = (g_1, \ldots, g_t),\ h = (h_1, \ldots, h_t)$ and $f = (f_1, \ldots, f_t)$.

3. Pick functions $\mu_i : M_i \mapsto \{\omega, \omega^2, \omega^3\}$ by defining independently for every $x \in M_i$

$$\mu_i(x) = \begin{cases} \omega & \text{with probability } 0.2 \\ \omega^2 & \text{with probability } 0.6 \\ \omega^3 & \text{with probability } 0.2 \end{cases}$$

   Let $\mu = (\mu_1, \ldots, \mu_t)$.

4. Define function $g'_i : M_i \mapsto \{1, \omega, \ldots, \omega^{p-1}\}$ by defining for every $x \in M_i$,

$$g'_i(x) = g_i(x) f_i(\pi^{v,w,i}(x))\ \mu_i(x)$$

   Let $g' = (g'_1, \ldots, g'_t)$. We will use the shortform $g' = g \cdot (f \circ \pi^{v,w}) \cdot \mu$ where $\circ$ denotes composition of functions.

5. Define functions $h'_i : M_i \mapsto \{1, \omega, \ldots, \omega^{p-1}\}$ by defining for every $x \in M_i$,

$$h'_i(x) = h_i(x) f_i(\pi^{v,w',i}(x))$$

   Let $h' = (h'_1, \ldots, h'_t)$. We will use the shortform $h' = h \cdot (f \circ \pi^{v,w'})$.

6. Accept if and only if

$$\text{Not-All-Equal}(B(g), B(g'), C(h), C(h'))$$

Picking the function μ is the most novel aspect of this construction (see soundness analysis and the proof of Lemma 5.4.3).

Completeness and the Number of Random Bits Used

It is easy to see that the test always accepts a correct proof. In a correct proof, B is the Split Code of the label of w, say $b = (b_1, \ldots, b_t)$ and C is the Split Code of the label of w', say $c = (c_1, \ldots, c_t)$. Also v has a label $a = (a_1, \ldots, a_t)$ and we have correct projections i.e. $a_i = \pi^{v,w,i}(b_i) = \pi^{v,w',i}(c_i)$ for $1 \le i \le t$. Hence $B(g) = g(b)$,

$$B(g') = g'(b) = g(b)f(\pi^{v,w}(b))\mu(b) = g(b)f(a)\mu(b),$$

$C(h) = h(c), C(h') = h(c)f(a)$. So the test will accept provided $\mu(b) \neq 1$, i.e. provided

$$\prod_{i=1}^{t} \mu_i(b_i) \neq 1$$

Note that μ_i takes values in the set $\{\omega, \omega^2, \omega^3\}$. Hence the left hand side is ω^r for some $t \le r \le 3t$. Since $p \ge 3t + 1$, the left hand side $\neq 1$.

Clearly the number of random bits used is at most $O(ut \log n + t\, 2^{O(u)} \log p)$.

5.4 Soundness Analysis

We prove the following theorem in this section which suffices to prove Theorem 5.1.1 with appropriate choice of parameters.

Theorem 5.4.1 *For any set $\mathcal{I}$ of δ fraction of vertices in the hypergraph, there are $\delta^4 - p^{O(t)}2^{-\gamma ut}$ fraction of edges which lie entirely in this set. In particular there is no independent set of size $2^{-\gamma ut/8}$ (think of t, p as constants and u as growing).*

Recall that $\mathcal{I}$ corresponds to a set of locations in the proof. Define a $0-1$ proof in the follows way : For $w \in W$, let B be the corresponding table. Define

$$B(g) = \left\{ \begin{array}{ll} 1 & \text{if the location } B(g) \in \mathcal{I} \\ 0 & \text{otherwise} \end{array} \right.$$

Note that the zero Fourier coefficient gives the average value of an encoding. Therefore $\widehat{B}_0$ is the fraction of 1s in table B. This implies that the fraction of 1s in the whole proof is $E_w[\widehat{B}_0]$ and since $\mathcal{I}$ contains a δ fraction of proof-locations, we have

$$E_w[\widehat{B}_0] = \delta \tag{5.1}$$

Consider the process of picking a random hyperedge, which corresponds to the choice of picking v, w, w', f, g, h, μ. This edge has 4 vertices $B(g), B(g'), C(h), C(h')$. All 4 vertices fall in the set $\mathcal{I}$ if and only if all 4 proof-locations contain bit-value 1. Therefore the fraction of edges contained entirely in set $\mathcal{I}$ is

$$E_{v,w,w',f,g,h,\mu}\big[B(g)B(g')C(h)C(h')\big]$$

We analyze this expression in the following. First we state a couple of easy lemmas which we prove in Section 5.5.

Definition 5.4.2 *For a function $\beta_i : M_i \mapsto GP(p)$, let $|\beta_i| = |\{x \in M_i : \ \beta_i(x) \neq 0\}|$.*

Lemma 5.4.3 *For $\beta = (\beta_1, \ldots, \beta_t)$,*

$$\begin{aligned} E_{\mu=(\mu_1,\ldots,\mu_t)}\left[\chi_\beta(\mu)\right] &= Q(\beta)\, \omega^{2\ \sum_{i=1}^{t} weight(\beta_i)} \\ &= Q(\beta)\, \omega^{2 \cdot weight(\beta)} \end{aligned}$$

where $Q(\beta)$ is a positive real number satisfying

$$Q(\beta) \leq (1 - \Omega(\frac{1}{p^2}))^{|\beta_1|+|\beta_2|+\dots|\beta_t|}$$

Lemma 5.4.4 *If B is a real-valued table, then $\widehat{B}_{-\beta} = \overline{\widehat{B}_{\beta}}$.*

Let us proceed with the analysis. Fix v, w, w' for the moment. Using Fourier expansions of B, C we get

$$E_{f,g,h,\mu}\big[\sum_{\phi,\beta,\psi,\gamma} \widehat{B}_\phi \widehat{B}_\beta \widehat{C}_\psi \widehat{C}_\gamma \;\; \chi_\phi(g)\chi_\beta(g(f\circ\pi^{v,w})\mu)\chi_\psi(h)\chi_\gamma(h(f\circ\pi^{v,w'}))\big]$$

where $\phi = (\phi_1, \phi_2, \dots, \phi_t), \phi_i : M_i \mapsto GF(p)$ and similarly for β, ψ, γ. Simplifying,

$$E_{f,g,h,\mu}\big[\sum_{\phi,\beta,\psi,\gamma} \widehat{B}_\phi \widehat{B}_\beta \widehat{C}_\psi \widehat{C}_\gamma \;\; \chi_{\phi+\beta}(g)\chi_{\psi+\gamma}(h)\chi_\beta(f\circ\pi^{v,w})\chi_\gamma(f\circ\pi^{v,w'})\;\chi_\beta(\mu)\big]$$

The expectation is zero unless $\phi = -\beta$ and $\psi = -\gamma$ where the equality holds on each of the t components. For function $\beta_i : M_i \mapsto GF(p)$ and the projection $\pi^{v,w,i} : M_i \mapsto N_i$, define $\pi_p^{v,w,i} : N_i \mapsto GF(p)$ as :

$$\pi_p^{v,w,i}(a) = \sum_{x\in M_i \,:\, \pi^{v,w,i}(x)=a} \beta_i(x) \quad \forall\, a \in N_i$$

Let $\pi_p^{v,w}(\beta) = (\pi_p^{v,w,1}(\beta_1), \pi_p^{v,w,2}(\beta_2), \dots, \pi_p^{v,w,t}(\beta_t))$. With this definition, it can be easily verified that

$$\chi_\beta(f\circ\pi^{v,w}) = \chi_{\pi_p^{v,w}(\beta)}(f), \qquad \chi_\gamma(f\circ\pi^{v,w'}) = \chi_{\pi_p^{v,w'}(\gamma)}(f)$$

Hence the expression reduces to

$$E_{f,\mu}\big[\sum_{\beta,\gamma} \widehat{B}_{-\beta}\widehat{B}_{\beta}\widehat{C}_{-\gamma}\widehat{C}_{\gamma}\ \ \chi_{\pi_p^{v,w}(\beta)+\pi_p^{v,w'}(\gamma)}(f)\ \chi_{\beta}(\mu)\big]$$

The expectation over f is zero unless $\pi_p^{v,w}(\beta) + \pi_p^{v,w'}(\gamma) = 0$. Lemma 5.4.3 gives the value of $E_{\mu}[\chi_{\beta}(\mu)]$. Also, note that B, C are $0-1$ tables, hence take only real values. Therefore we can use Lemma 5.4.4 and get

$$E_{v,w,w'}\left[\sum_{\beta,\gamma:\pi_p^{v,w}(\beta)+\pi_p^{v,w'}(\gamma)=0} |\widehat{B}_{\beta}|^2|\widehat{C}_{\gamma}|^2 Q(\beta)\omega^{2\cdot weight(\beta)}\right]$$

Note that when $weight(\beta) = 0$, the corresponding terms are non-negative real numbers. Among these terms we retain only the term with $\beta = \gamma = 0$. Thus the expression is at least,

$$E_{v,w,w'}\left[|\widehat{B}_0|^2|\widehat{C}_0|^2\right] - E_{v,w,w'}\left[\sum_{\beta,\gamma:\pi_p^{v,w}(\beta)=-\pi_p^{v,w'}(\gamma),weight(\beta)\neq 0} |\widehat{B}_{\beta}|^2|\widehat{C}_{\gamma}|^2 Q(\beta)\right]$$

Note that $\widehat{B}_0, \widehat{C}_0$ are non-negative reals and

$$Q(\beta) \leq \prod_{i=1}^{t}(1-\Omega(\frac{1}{p^2}))^{|\beta_i|} \leq \prod_{i=1}^{t} O(p^2)\frac{1}{|\beta_i|}$$

Thus the above expression is at least

$$E_{v,w,w'}\left[\widehat{B}_0^2\widehat{C}_0^2\right] - p^{O(t)}E_{v,w,w'}\left[\sum_{\beta,\gamma:\pi_p^{v,w}(\beta)=-\pi_p^{v,w'}(\gamma),weight(\beta)\neq 0} |\widehat{B}_{\beta}|^2|\widehat{C}_{\gamma}|^2 \prod_{i=1}^{t}\frac{1}{|\beta_i|}\right]$$

We will show that this expression is at least $\delta^4 - p^{O(t)}2^{-\gamma ut}$. We observe that the first term corresponds to the fraction of 1-bits in the proof and Lemma 5.4.5 shows that it is at least δ^4. The second term can be used to extract a labeling for the Split Label Cover problem and Lemma 5.4.6 shows that it is at most $p^{O(t)}2^{-\gamma ut}$.

Lemma 5.4.5 $E_{v,w,w'}[\widehat{B}_0^2\widehat{C}_0^2] \geq \delta^4$

Proof: Note that after fixing v, we pick w, w' independently with identical distribution. Hence

$$E_{v,w,w'}[\widehat{B}_0^2\widehat{C}_0^2] = E_v\left[\left(E_w\left[\widehat{B}_0^2\right]\right)^2\right] \geq \left(E_{v,w}\left[\widehat{B}_0^2\right]\right)^2 = \left(E_w\left[\widehat{B}_0^2\right]\right)^2$$
$$\geq \left(\left(E_w\left[\widehat{B}_0\right]\right)^2\right)^2 \quad = \quad \delta^4$$

using Equation (5.1). ■

Lemma 5.4.6 *There is a labeling for the Split Label Cover problem satisfying at least the following fraction of edges :*

$$E_{v,w,w'}\left[\sum_{\beta,\gamma:\pi_p^{v,w}(\beta)=-\pi_p^{v,w'}(\gamma),weight(\beta)\neq 0} |\widehat{B}_\beta|^2|\widehat{C}_\gamma|^2\prod_{i=1}^{t}\frac{1}{|\beta_i|}\right]$$

In particular, in soundness case, this expression is at most $2^{-\gamma ut}$, i.e. the soundness parameter of the Split Label Cover problem.

Proof: For every vertex $w \in W$, we pick $\beta = (\beta_1, \ldots, \beta_t)$ with probability $|\widehat{B}_\beta|^2$ and define label of w to be $(b_1, \ldots, b_t)$ where each b_i is a randomly chosen element $x_i \in M_i$ for which $\beta_i(x_i) \neq 0$. For every vertex $v \in V$, we pick its random neighbor $w' \in W$, pick

$\gamma = (\gamma_1, \dots, \gamma_t)$ with probability $|\widehat{C}_\gamma|^2$, define c_i to be a randomly chosen element $y_i \in M_i$ for which $\gamma_i(y_i) \neq 0$ and finally define label of v to be $(\pi^{v,w',1}(c_1), \dots, \pi^{v,w',t}(c_t))$.

We claim that the fraction of edges satisfied by this labeling is at least the expectation in the lemma. Note that the summation runs only over β, γ such that $weight(\beta) \neq 0$. The tables are equality folded. Thus Lemma 5.2.7 implies that for all i, $\beta_i \neq 0, \gamma_i \neq 0$. Hence there always exists $x_i \in M_i$ such that $\beta_i(x_i) \neq 0$ and $y_i \in M_i$ such that $\gamma_i(y_i) \neq 0$. Also, with probability $\prod_{i=1}^{t} \frac{1}{|\beta_i|}$, it holds that

$$\pi^{v,w,i}(b_i) = \pi^{v,w',i}(c_i) \ \ \forall\, i$$

In other words $\pi^{v,w}(b_1, \dots, b_t) = \pi^{v,w'}(c_1, \dots, c_t)$, i.e. $\pi^{v,w}(\text{label}(w)) = \text{label}(v)$ which is precisely the condition for satisfying an edge (v, w). Hence the claim follows. ■

5.5 Proofs of Lemmas 5.4.3, 5.4.4 and Theorem 5.1.1

Proof of Lemma 5.4.3

Let us first prove the following lemma.

Lemma 5.5.1 *Let $\beta_i : M_i \mapsto GF(p)$ and pick function $\mu_i : M_i \mapsto \{\omega, \omega^2, \omega^3\}$ where for every $x \in M_i$, we define $\mu_i(x)$ to be ω with probability 0.2, ω^2 with probability 0.6 and ω^3 with probability 0.2. Then*

$$E_{\mu_i}[\prod_{x \in M_i} \mu_i(x)^{\beta_i(x)}] = Q(\beta_i)\ \omega^{2 \cdot weight(\beta_i)}$$

where $Q(\beta_i)$ is a positive real number satisfying $Q(\beta_i) \leq (1 - \Omega(\frac{1}{p^2}))^{|\beta_i|}$.

Proof: We note that the given expectation is

$$\prod_{x \in M_i} E_{\mu_i}[\mu_i(x)^{\beta_i(x)}]$$

We show that for every x such that $\beta_i(x) \neq 0$, the inner expectation is $\omega^{2 \cdot \beta_i(x)}$ times a positive number $\leq 1 - \Omega(\frac{1}{p^2})$. Let $r = \beta_i(x)$. Because of the way $\mu_i()$ is defined, the inner expectation is

$$0.2\, \omega^r + 0.6\, \omega^{2r} + 0.2\, \omega^{3r} = \omega^{2r}(0.6 + 0.2\, \omega^r + 0.2\, \omega^{-r})$$

This complex number is ω^{2r} times a positive real number which is maximized when $r = 1$ and this maximum value is $0.6 + 0.4 \cos(\frac{2\pi}{p}) = 1 - \Omega(\frac{1}{p^2})$. ■

Now we prove Lemma 5.4.3. Note that

$$\begin{aligned} E_\mu\left[\chi_\beta(\mu)\right] &= E_\mu\left[\prod_{i=1}^{t} \prod_{x_i \in M_i} \mu_i(x_i)^{\beta_i(x_i)}\right] \\ &= \prod_{i=1}^{t} E_{\mu_i}\left[\prod_{x_i \in M_i} \mu_i(x_i)^{\beta_i(x_i)}\right] \end{aligned}$$

Lemma 5.4.3 now follows from Lemma 5.5.1, with

$$Q(\beta) = Q(\beta_1)Q(\beta_2)\cdots Q(\beta_t)$$

Proof of Lemma 5.4.4

Since B is real-valued,

$$\widehat{B}_{-\beta} = E_g\big[B(g)\overline{\chi_{-\beta}(g)}\big] = E_g\big[\overline{B(g)\chi_{-\beta}(g)}\big] = E_g\big[\overline{B(g)}\chi_\beta(g)\big] = \overline{E_g\big[B(g)\overline{\chi_\beta(g)}\big]} = \overline{\widehat{B}_\beta}$$

Proof of Theorem 5.1.1

We have $t = \lfloor \frac{p-1}{3} \rfloor$. Let $u = O(\log \log n)$. The number of random bits used by the verifier is

$$O(ut \log n + t\, 2^{O(u)} \log p) \leq 2^{O(u)}$$

The hypergraph constructed has size N which is at most exponential in the number of random bits used by the verifier, so $\log N \leq 2^{O(u)}$.

We know that in completeness case, the hypergraph is p-colorable. By Theorem 5.4.1, in soundness case, there is no independent set of size $2^{-\gamma ut/8} = (\log N)^{-cp}$ for some absolute constant $c > 0$.

Chapter 6

Hardness of Shortest Vector Problem in High L_p Norms

Shortest Vector Problem (SVP) is one of the most important problems in complexity theory. Given a basis for an n-dimensional lattice, the goal is to find the shortest non-zero vector in the lattice. SVP has applications to relationship between the worst-case and average-case complexity of problems, breaking and building cryptosystems (!), factoring rational polynomials and numerous other areas in mathematics and computer science. Characterizing the hardness of SVP is a major open problem. The famous LLL algorithm [87] achieves an exponential approximation factor for SVP whereas even NP-hardness wasn't known until recently.

In this chapter, we obtain a big improvement in the hardness results known for SVP in high L_p norms (Theorem 6.1.1). Apart from the improved hardness factor, our reduction is much simpler and direct, much more *elementary*, and holds under a weaker complexity assumption. We believe that our ideas could be applicable in getting any constant factor hardness in some fixed L_p (maybe even in L_2) norm.

6.1 Result and History of the Problem

An n-dimensional lattice $\mathcal{L}$ is a set of vectors $\{\sum_{i=1}^{n} a_i v_i \mid a_i \in \mathbf{Z}\}$ where $v_1, v_2, \ldots, v_n \in \mathbf{R}^m$ is a set of independent vectors called the basis for the lattice. The same lattice could have many bases. Given a basis for an n-dimensional lattice, the Shortest Vector Problem asks for the shortest non-zero vector in the lattice. The length of the vectors can be measured in any L_p norm ($p \geq 1$) and the corresponding problem is denoted by SVP$_p$. This problem has a beautiful history and we present some of the results below. For a more comprehensive list of references and a thorough treatment of the subject, we refer to Micciancio and Goldwasser's book [97]. We also recommend Micciancio's PhD thesis [95] and an expository article by Kumar and Sivakumar [82].

The Shortest Vector Problem has been studied since the time of Gauss ([49], 1801) who gave an algorithm for SVP$_2$ in 2-dimensions. The general problem for arbitrary dimensions was formulated by Dirichlet in 1842. The theory of Geometry of Numbers by Minkowski [98] deals with the existence of shortest non-zero vectors in lattices. In a celebrated result, Lenstra, Lenstra and Lovász [87] gave a polynomial time algorithm for approximating SVP$_2$ within factor $2^{n/2}$. This algorithm has numerous applications, e.g. factoring rational polynomials [87], breaking knapsack-based codes [84], checking the solvability by radicals [85] and integer programming in a fixed number of variables ([87], [88], [68]). Schnorr [107] improved the approximation factor to $2^{\delta n}$ for any $\delta > 0$. Since all L_p norms are within factor $\sqrt{n}$ from the L_2 norm, these algorithms give similar approximations for SVP$_p$ for any p. It is a major open problem whether SVP has polynomial factor approximations that run in polynomial time. Exact computation of SVP$_2$ in exponential time has also been investigated, for instance [69], [4].

In 1981, van Emde Boas [112] proved that SVP$_\infty$ is NP-hard and conjectured that the same is true for any L_p norm. However proving NP-hardness for any finite p (in particular $p = 2$) was an embarrassing open problem for long time. A breakthrough result by Ajtai [2] in 1998 finally showed that SVP$_2$ is NP-hard under randomized reductions. Cai and Nerurkar [25] improved Ajtai's result to a hardness of approximation result showing a hardness factor of $(1 + \frac{1}{n^\delta})$. Another breakthrough by Micciancio [96] showed that SVP$_p$ is hard to approximate within factor $2^{1/p} - \delta$ for every $\delta > 0$.

Showing hardness of approximation results for SVP was greatly motivated by Ajtai's discovery [1] of worst-case to average-case hardness and subsequent construction of lattice-based public key cryptosystem by Ajtai and Dwork [3]. Ajtai showed that if there is a randomized polynomial time algorithm for solving SVP$_2$ on a non-negligible fraction of lattices from a certain natural class of lattices, then there is a randomized polynomial time algorithm for approximating SVP$_2$ on *every* instance within some polynomial factor n^c. Ajtai-Dwork's work gave hope, for the first time, that cryptography could be based on the (conjectured) worst-case hardness of a problem. Their work implies that if n^c-approximation to SVP$_2$ is hard, then one can construct a secure cryptosystem. The constant c was noted to be 19 in [23], and brought down to $9 + \delta$ by Cai and Nerurkar [24] and then to $4 + \delta$ by Cai [23]. Recently, Regev [104] gave an alternate construction of a public key cryptosystem based on $n^{1.5}$-hardness of SVP$_2$ (actually a variant called unique-SVP$_2$) ! Unfortunately, there are barriers to showing such strong hardness results. In fact, showing factor n NP-hardness would imply that NP = coNP [83] and showing factor $\sqrt{n/O(\log n)}$ NP-hardness would imply that coNP $\subseteq$ AM [52].

Another related problem that has received much attention is the Closest Vector Problem (denoted CVP$_p$) where given a lattice and a vector y, the problem is to find the lattice vector that is closest to y. In spite of the apparent similarity between SVP and CVP, they turn out to be quite different problems. Indeed, CVP$_p$ was shown to be NP-hard for all $p \geq 1$ by van Emde Boas [112]. Arora, Babai, Sweedyk, and Stern [9] used the PCP machinery to show that approximating CVP$_p$ (for all $p \geq 1$) within factor $2^{\log^{1-\delta} n}$ is hard unless NP $\subseteq$ DTIME($n^{poly(\log n)}$). This was improved to a NP-hardness result by Dinur, Kindler, and Safra [30] (their result even gives a subconstant value of δ, i.e. $\delta = (\log \log n)^{-c}$ for any $c < \frac{1}{2}$). Incidently, SVP$_\infty$ seems to behave very much like CVP$_\infty$, Dinur [27] shows factor $n^{1/\log \log n}$ NP-hardness for both these problems. Thus for SVP, the cases $p = \infty$ and $p < \infty$ seem to be qualitatively different.

Our Result

In this chapter, we obtain an improved hardness result for SVP$_p$ for large (but finite) values of p. Specifically we show that

Theorem 6.1.1 *For every $\epsilon > 0$, there is a constant $p(\epsilon)$ such that for all integers $p \geq p(\epsilon)$, it is NP-hard to approximate SVP$_p$ within factor $p^{1-\epsilon}$ under randomized reductions.*

This improves the hardness factor $2^{1/p} - \delta$ by Micciancio [96] for all large values of p. The result however is only asymptotic and says nothing about small values of p. The value $p(\epsilon)$ depends on a non-explicit constant in Raz's Parallel Repetition Theorem [103] (the constant γ in Theorem 2.3.2).

Significance of the result : Apart from the improved hardness factor, we believe that the result is significant in the following aspects :

1. Very strong hardness results are known for SVP$_\infty$, so it is reasonable to expect that the hardness result gets better as p grows. Our result ($p^{1-\epsilon}$) supports this intuition, whereas Micciancio's result ($2^{1/p} - \delta$) goes in the other direction.

2. Our result is a direct reduction from 2-Prover Games (or equivalently the Label Cover problem). Micciancio's reduction can be seen as an indirect reduction from 2-Prover Games. Lund and Yannakakis [93] reduce 2-Prover Games to the Set Cover problem. Arora et al [9] reduce Set Cover to CVP$_p$ for every finite value of p. Micciancio reduces CVP$_p$ to SVP$_p$ by constructing a sophisticated lattice as a gadget. Constructing this lattice requires many ideas from Ajtai's reduction. In contrast to our and Micciancio's reductions, Ajtai uses a reduction from a much more elementary problem, namely Subset Sum.

3. Our result holds under the assumption that NP $\not\subseteq$ ZPP. Ajtai's and Micciancio's reductions require a stronger assumption that NP $\not\subseteq$ BPP. Our reduction is considerably simpler and might be easier (if possible at all) to derandomize.

On the flip-side, our result doesn't apply for any small explicit value of p. Definitely, the most interesting case is L_2 norm, since Ajtai's worst-case to average-case reduction is based on hardness of SVP$_2$. However, we think that our result helps in a better understanding of the Shortest Vector Problem. Considering that even NP-hardness wasn't known for any finite L_p norm till 1998, it is interesting that we give a simple and straightforward proof.

6.2 Problem Definition and Techniques

We prove Theorem 6.1.1 via a reduction from the Label Cover problem given by Theorem 2.3.2. This is a direct reduction without using long code and with no Fourier analysis. The reduction is quite simple and the basic idea appears in Section 6.3.

We first redefine the Shortest Vector Problem in a different (but equivalent) manner, without any reference to lattices.

Problem Definition

The problem SVP$_p$ is defined as follows : Given a vector $\mathbf{x}$ of integer variables $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ and linear forms $\{\phi_1, \phi_2, \ldots, \phi_m\}$ where

$$\phi_i = \sum_{j=1}^{n} b_{ij} x_j \qquad b_{ij} \in \mathbb{R}$$

The goal is to find a non-zero integer vector $\mathbf{x}$ which minimizes the following objective function :

$$\text{OBJ} = \sum_{i=1}^{m} |\phi_i(\mathbf{x})|^p$$

Remarks : (1) Think of the basis vectors in the lattice as the columns of the matrix $\{b_{ij}\}$ and think of the integer variables x_is as the (unknown) coefficients when the shortest vector is written as an integer linear combination of the basis vectors. (2) We actually want to minimize OBJ$^{1/p}$. In order to show a factor k-hardness for SVP$_p$, it suffices to show a factor k^p-hardness for the above objective function.

A Nice Technique

Here we describe one of the techniques used in this chapter which could be of independent interest.

A common problem encountered in showing hardness of SVP is the following : We desire a reduction from some NP-hard problem, say Label Cover, Set Cover or CVP. Usually, it is straightforward to construct a set of vectors $\{v_1, v_2, \ldots, v_m\}$ which one could potentially use as the basis vectors for an instance of SVP. For completeness, there is a non-zero integer linear combination $\sum_{i=1}^{m} x_i v_i$ with short length. This combination corresponds to a correct labeling to Label Cover or a solution to the Set Cover instance depending on what problem we started with. The combination typically has the property that δm of the coefficients x_is are non-zero. Now let us say we want to show the soundness property, i.e. if the NP-hard problem we started with is a NO instance, then there is no short non-zero integer linear combination. However, typically it so happens that each of the vectors v_i itself is a short vector. Thus for any j, setting $x_j = 1$ and $x_i = 0$ for $i \neq j$ gives a short lattice vector. In general, setting *too few* of x_is to non-zero values produces a short vector. We wish to somehow enforce the condition that many of the x_is must be set to a non-zero value.

We do this by augmenting the vectors v_is by one extra co-ordinate a_i. Call these augmented vectors $v'_i = (v_i, a_i)$ and let them be the basis vectors for an SVP instance. The set of integers $\{a_i : 1 \leq i \leq m\}$ satisfies :

- For any set $Y \subseteq [m],\ |Y| = \delta m$, the integers $\{a_j | j \in Y\}$ have a non-zero $\{0, 1, -1\}$-linear combination that vanishes.

- For any set $Z \subseteq [m],\ |Z| < \frac{\delta m}{20 \log(1/\delta)}$, a non-zero $\{0, 1, -1\}$-linear combination of integers $\{a_j | j \in Z\}$ cannot vanish.

It can be shown that choosing m random integers from the range $[1, 2, \ldots, 2^{\delta m/2}]$ satisfies these properties with high probability.

For completeness, since δm of the x_is are non-zero, one could hope to set them to appropriate $\{0, 1, -1\}$ values so that $\sum_{i=1}^{m} x_i a_i = 0$ and the vector $\sum_{i=1}^{m} x_i v_i$ is short.

For soundness, assume for the moment that x_is are restricted to take values $\{0, 1, -1\}$. It is clear that if at most $\frac{\delta m}{20 \log(1/\delta)}$ of the x_is are non-zero, then $\sum_{i=1}^{m} x_i a_i$ cannot vanish. One can apply a huge penalty if this sum (which is the last co-ordinate of the linear combination $\sum_{i=1}^{m} x_i v'_i$) doesn't vanish. Thus, we are able to enforce the constraint that one must set at least $\frac{\delta m}{20 \log(1/\delta)}$ of the x_is to non-zero value. In general, x_is could take arbitrary integer values (not just $\{0, 1, -1\}$), but this can be handled as well, as we will see.

Construction of the set $\{a_i | 1 \leq i \leq m\}$ is the only place where our reduction is randomized. We would like to remark that Micciancio [96] also needs a gadget to enforce a similar condition and almost all work in his paper is devoted to constructing this gadget. More specifically, his gadget is a sophisticated lattice and he needs to enforce the condition that *one* particular coefficient x_{i_0} in the integer linear combination is set to a non-zero value.

The Label Cover Problem

The reduction is from the Label Cover problem. We need some more notation and an extra regularity property for Label Cover instances. So we restate Theorem 2.3.2 as

Theorem 6.2.1 *There is an absolute constant $\gamma > 0$ such that for every integer parameter u, it is NP-hard to distinguish between the following two cases : A Label Cover problem $\mathcal{L}(G(V, W, E), N, M, \{\pi^{v,w} | (v, w) \in E\})$ has*

- $OPT(\mathcal{L}) = 1$ *OR*
- $OPT(\mathcal{L}) \leq 1/R^{\gamma}$ *where $R = |M|$*

We denote $M = [R], N = [S], n = |V|, m = |W|$ and D to be the degree of every vertex in V. It can be assumed that $R = 6^u$, $S = 3^u$, $D = 5^u$ and $m = (5/2)^u n$. Moreover, for every edge (v, w), the map $\pi^{v,w} : [R] \mapsto [S]$ is "regular", meaning for every $j \in [S]$, there are exactly R/S elements in $[R]$ that are mapped to j.

Proof: As shown in [39], there is $\theta > 0$ such that it is NP-hard to tell whether a 5-regular graph is 3-colorable or no coloring of its vertices with 3 colors can make $1 - \theta$ fraction of the edges non-monochromatic. Starting with this gap-problem, one can construct a 2-Prover-1-Round game as follows : Pick a random edge (x, y) of the graph and one of its endpoints at random (say x). Ask prover P_2 for coloring of the vertices (x, y), his answer is supposed to be one of the 6 valid (non-monochromatic edge-) colorings. Ask prover P_1 for the color of x, his answer is supposed to be one of the 3 colors. Accept if and only if answers of P_1 and P_2 agree on color of x. It is easy to see that this game has soundness strictly less than 1 and applying Parallel Repetition Theorem gives an instance of Label Cover with all the properties listed above. ■

6.3 The Basic Idea in the Reduction

Here we describe the basic idea of the reduction. One needs to fix a lot of technical details later and we present a complete reduction in Section 6.4. Theorem 6.2.1 gives an instance of the Label Cover problem specified as

$$(G(V, W, E), [S], [R], \{\pi^{v,w}\}), \qquad n = |V|, m = |W|, \quad D = \text{degree}(v)\ \forall\ v \in V$$

The intended hardness factor we wish to achieve for SVP$_p$ is $k^{1-\epsilon}$. We will have $k = R^{\gamma/20}$ and $p = O(k)$. The integer variables of the SVP$_p$ will be

$$\{x_{w,i} \mid w \in W, i \in [R]\ \}$$

For a fixed w, let $B(w)$ be the “block” of variables defined as

$$B(w) = \{x_{w,i} \mid i \in [R]\ \}$$

For a vertex $v \in V$, let $N(v) \subseteq W$ denote the set of neighbors of v with $|N(v)| = D$. There will be one linear form in SVP$_p$ for every $v \in V$ and every sequence $(w_1, w_2, \ldots, w_S)$ where $w_j \in N(v)$ for $1 \leq j \leq S$. The linear form is sum of R variables, with R/S variables each from the block $B(w_j)$. The linear form is defined to be :

$$\sum_{j=1}^{S} \sum_{i \in [R] : \pi^{v,w_j}(i) = j} x_{w_j,i} \tag{6.1}$$

Note that the total number of linear forms is nD^S.

Completeness

For completeness, we will show that if there is a labeling A for the label cover problem satisfying every edge (i.e. $OPT(\mathcal{L}) = 1$), then the problem SVP$_p$ has a solution with the objective function OBJ $= nD^S$. Let $A : W \mapsto [R],\ A : V \mapsto [S]$ be such a labeling. For every edge (v, w) in the Label Cover instance, we have $\pi^{v,w}(A(w)) = A(v)$.

Define a solution as

$$x_{w,i} = \begin{cases} 1 & \text{if} \quad A(w) = i \\ 0 & \text{otherwise} \end{cases}$$

Note that there are nD^S linear forms of type (6.1). We will show that each of these linear forms equals 1. Note that out of all the variables in the linear form (6.1), exactly one equals 1 and the rest are all 0. The non-zero variable is $x_{w_j,i}$ for which $j = A(v), i = A(w_j)$ (Verify ! This is the crux of the reduction).

Soundness

We wish to show that if OPT $< 1/R^\gamma$ for the Label Cover problem, then for any non-zero integer vector $\mathbf{x} = \{x_{w,i}\}$, the objective function OBJ is at least $\frac{1}{2}nD^{S-k}k^p$. We will show this only for a restricted class of vectors, i.e. vectors $\mathbf{x} = \{x_{w,i}\}$ which "arise" out of labelings $A : W \mapsto [R]$. Eventually we want this to work for **every** non-zero vector $\{x_{w,i}\}$. This involves a lot of technical details and we present the full reduction in Section 6.4. So let us assume $A : W \mapsto [R]$ is an assignment and $\{x_{w,i}\}$ is the corresponding vector, i.e.

$$x_{w,i} = \begin{cases} 1 & \text{if} \quad A(w) = i \\ 0 & \text{otherwise} \end{cases}$$

For every vertex $v \in V$, define a set of labels $\Psi(v) \subseteq [S]$ as follows :

$$\Psi(v) = \{\pi^{v,w}(A(w)) \mid w \in N(v)\ \}$$

Lemma 6.3.1 *For at least half the vertices in V, $|\Psi(v)| \geq k$ (call such vertices "good").*

Proof: Assume on the contrary that half the vertices in V have $|\Psi(v)| \leq k$. Define label for vertex v to be a random element of $\Psi(v)$. Note that for every $w \in N(v)$, $\pi^{v,w}(A(w)) \in \Psi(v)$. Therefore with probability $1/k$, the edge (v, w) is satisfied by this random labeling to v. Hence there exists a labeling for the Label Cover problem that satisfies at least a fraction $\frac{1}{2k}$ of the edges. This is a contradiction since $\frac{1}{2k} > \frac{1}{R^\gamma}$ (recall that $k = R^{\gamma/20}$). ■

For every good vertex v, assume w.l.o.g. that $\{1, 2, \ldots, k\} \subseteq \Psi(v)$. Thus for every $1 \leq j \leq k$, there exists $w_j^* \in N(v)$ such that $\pi^{v,w_j^*}(A(w_j^*)) = j$. Hence for any sequence $(w_1^*, w_2^*, \ldots, w_k^*, w_{k+1}, \ldots, w_S)$ where $w_{k+1}, \ldots, w_S \in N(v)$ are arbitrary, the linear form (6.1) is at least equal to k. This contributes k^p to the objective function OBJ.

Half the vertices are good, hence OBJ $\geq \frac{n}{2} D^{S-k} k^p$.

Hardness Factor

Note that the objective function is nD^S in completeness case and at least $\frac{1}{2} nD^{S-k} k^p$ in the soundness case. Choose p such that

$$\frac{1}{2} nD^{S-k} k^p \; > \; nD^S k^{(1-\epsilon)p}$$

Thus there is a penalty of a factor $k^{(1-\epsilon)p}$ in the soundness case. Noting that $D \leq R, k = R^{\gamma/20}$, we see that it suffices to take $p = \frac{20k}{\epsilon\gamma}$. This gives hardness factor of $k^{1-\epsilon}$ or $p^{1-\epsilon'}$ as desired. This completes the basic idea in the reduction. We give the full reduction in the next section.

6.4 Full Reduction

We again fix $k = R^{\gamma/20}$ and choose $p = O(k)$ later. The intended hardness factor we wish to achieve for SVP$_p$ is $k^{1-\epsilon}$. The set of integer variables is the same, namely

$$\mathbf{x} = \{x_{w,i} \mid w \in W, i \in [R]\}$$

There will be 4 types of linear forms. These forms are supposed to "handle" different types of non-zero vectors $\mathbf{x}$ in the soundness case. The exact role of these linear forms will be clear as we go along.

Type-1 linear forms :

$$\forall\, w \in W,\ \forall i\ \in [R], \qquad x_{w,i}$$

Thus the Type-1 linear forms are just the variables themselves.

Type-2 linear form :

$$\sum_{w \in W, i \in [R]} a_{w,i} x_{w,i}$$

Thus there is only one Type-2 linear form. The coefficients $a_{w,i}$ in this linear form are randomly chosen integers from the range $[1, 2, ..., 2^{m/2}]$. Introducing this linear form is precisely the technique described in Section 6.2. We want to ensure that for soundness, one must set "many" of the variables $x_{w,i}$s to a non-zero value.

Type-3 linear forms :

$$\forall\, w \in W, \qquad \sum_{i=1}^{R} \pm x_{w,i}$$

Thus there are 2^R Type-3 linear forms for every $w \in W$. There are R variables in every form and there is one linear form for every choice of $+/-$ sign.

Type-4 linear forms :

$$\forall\, v \in V,\ \forall w_1, w_2, \ldots, w_S \in N(v), \qquad \sum_{j=1}^{S} \sum_{i \in [R] : \pi^{v,w_j}(i) = j} \pm x_{w_j,i}$$

There are $2^R D^S$ linear forms for every $v \in V$. These are essentially the linear forms used in Section 6.3, except that now we take all the $+/-$ combinations. Note that there are R variables in each form.

In the completeness case, Type-2 form will contribute 0. Type-1, Type-3 and Type-4 will contribute (at most) $m, 2^R m$ and $2^R n D^S$ respectively. We multiply the Type-1, Type-3 and Type-4 forms by appropriate quantities C_1, C_3, C_4 so that they contribute equally towards the objective function. More precisely, C_1, C_3, C_4 are chosen so that

$$C_1^p\, m = C_3^p\, 2^R m = C_4^p\, 2^R n D^S$$

In the soundness case, we will show that any non-zero vector $\mathbf{x} = \{x_{w,i}\}$ either produces a non-zero value in Type-2 form or it pays a penalty of factor $k^{(1-\epsilon)p}$ for at least one of the remaining three types.

We multiply Type-2 form by a huge quantity. Thus we incur a huge penalty unless the Type-2 form vanishes (it will vanish in the completeness case, so we are fine).

Completeness

In the completeness case, let $A : W \mapsto [R], A : V \mapsto [S]$ be a correct labeling. Let

$$x_{w,i} = \begin{cases} 0, 1 \text{ or } -1 & \text{if} \quad A(w) = i \\ 0 \quad \text{otherwise} \end{cases}$$

The choice of $0, 1, -1$ for the variables $\{x_{w,A(w)}\}$ is made such that the Type-2 form vanishes. We use the Pigeon-Hole principle (suggested by Sanjeev Arora). Consider the set of m variables $\{x_{w,A(w)}\}$, and the corresponding coefficients $a_{w,A(w)}$ in the Type-2 form. Consider the 2^m different sums for all subsets of these m coefficients. These sums take integer values in the range $[1, 2, 3, \ldots, m2^{m/2}]$. Hence sums for two distinct subsets must be equal, which gives a vanishing $\{0, 1, -1\}$ linear combination of the integers $a_{w,A(w)}$.

Now we look at the remaining 3 types.

- For Type-1 forms, note that at most m of the variables are ± 1, the rest are 0. So we get contribution of at most m.
- For Type-3 forms, note that for every w, there is at most one variable $x_{w,i}$ that is ± 1. Thus we get contribution of at most $2^R m$.
- Every Type-4 form is ± 1 as seen for completeness part in Section 6.3 and it might even be 0 since we "turn off" some of the variables $x_{w,A(w)}$ to 0. Thus the Type-4 forms contribute at most $2^R n D^S$.

6.5 Soundness of the Reduction

The crux of the soundness analysis is as in Section 6.3. However, we have to handle cases when $x_{w,i}$s are negative, or very few of them are non-zero. We do this in several stages.

For a vector $\mathbf{x} = \{x_{w,i}\}$, let $\#\mathbf{x}$ denote the number of variables (or coordinates) that are non-zero. For a block of variables $B(w)$, let $\#B(w)$ denote the number of non-zero variables in this block.

Handling $\mathbf{x}$ with $\#\mathbf{x} \leq \frac{m}{20 \log R}$ and $\|\mathbf{x}\|_1 \geq mR$

Such $\mathbf{x}$ are handled by Type-1 forms. Note that when at most $\frac{m}{20 \log R}$ coordinates are non-zero and the L_1 norm is at least mR, then L_p norm is minimized when all the non-zero coordinates are equal to $\frac{mR}{m/(20 \log R)}$. Hence

$$\sum_{w,i} |x_{w,i}|^p \geq (20 R \log R)^p \frac{m}{20 \log R} \geq R^{p/2} m \geq k^p m$$

Note that the contribution of Type-1 forms in the completeness case is at most m. Thus we get a penalty of factor k^p for the Type-1 forms.

Handling $\mathbf{x}$ with $\#\mathbf{x} \leq \frac{m}{20\log R}$ and $\|\mathbf{x}\|_1 \leq mR$

These are handled by Type-2 linear form. We show that with high probability, the Type-2 linear form *does not* vanish for any such $\mathbf{x}$.

The coefficients in this linear form are randomly chosen integers from the range $[1, 2, 3, \ldots, 2^{m/2}]$. Hence for any non-zero vector $\mathbf{x}$, the probability that the Type-2 form vanishes is at most $\frac{1}{2^{m/2}}$. We count the number of vectors $\mathbf{x}$ such that $\#\mathbf{x} \leq \frac{m}{20\log R}$ and $\|\mathbf{x}\|_1 \leq mR$. We show that there aren't too many of them and we can take a union bound. Number of such $\mathbf{x}$'s can be bounded by

$$\binom{mR}{\frac{m}{20\log R}} \cdot 2^{m/(20\log R)} \cdot \Big(\#\text{ non-negative integer solutions to}: \; y_1 + y_2 + \ldots + y_{\frac{m}{20\log R}} \leq mR\Big)$$

$$\leq \binom{mR}{\frac{m}{20\log R}} \cdot 2^{m/(20\log R)}\, mR \cdot \binom{2mR}{\frac{m}{20\log R}} \leq 2^{2m/5} \ll 2^{m/2}$$

Where one uses the fact that

$$\binom{M}{\delta M} \leq 2^{H(\delta)M} \leq 2^{2\delta\log(1/\delta)M}$$

Avoiding the Problem of Negative or Large Values

In general the variables $x_{w,i}$ could be positive or negative and could take large integer values. We will now show that one can as well assume that they take values only 0 or

1. This is done by averaging over all the $+/-$ linear combinations and this is the reason why Type-3 and Type-4 forms appear with all possible $+/-$ combinations.

Lemma 6.5.1 *Let $x_1, x_2, \ldots, x_t$ be t non-zero integers. Let $a_i \in \{1, -1\}$ be chosen randomly. Assume p to be an even integer. Then*

$$E_{a_1,a_2,\ldots,a_t}\left[|\sum_{i=1}^{t} a_i x_i|^p\right] \geq \max\{t^{p-t}, t^{p/2}\}$$

Proof: We can expand out the product $(\sum_{i=1}^{t} a_i x_i)^p$ and take the expectation of each term separately. For the terms in which some $a_i x_i$ occurs to an odd power, the expectation is zero. For the remaining terms, the expectation is at least 1. Thus the expectation is at least the number of terms such that every $a_i x_i$ occurs to an even power. In other words, we want to count the number of functions $f : [p] \mapsto [t]$ such that every $j \in [t]$ has even number of pre-images. Considering functions where $f(1) = f(2), f(3) = f(4), \ldots, f(p-1) = f(p)$, we get a bound of $t^{p/2}$. Another way is to take an arbitrary function $g : [p-t] \mapsto [t]$ and then "extend" it to a function $\tilde{g} : [p] \mapsto [t]$ where the values $\tilde{g}(p-t+1), \tilde{g}(p-t+2), \ldots, \tilde{g}(p)$ are chosen to make sure that for $\tilde{g}$, every value has an even number of pre-images. This gives a bound of t^{p-t}. ■

Lemma 6.5.1 implies that for any set of R variables, with at least t of them non-zero, when summed over all $+/-$ combinations, the contribution to the objective function OBJ is at least $2^R \max\{t^{p-t}, t^{p/2}\}$.

Handling One More Annoying Case

One more annoying case is when most of the non-zero variables belong to blocks $B(w)$ such that these blocks themselves have too many non-zero variables. To be precise, we want to avoid the situation where

$$\sum_{B(w):\#B(w)\geq k^3} \#B(w) \;\geq\; \frac{m}{40\log R} \tag{6.2}$$

This case is handled by Type-3 linear forms. By Lemma 6.5.1, for any block $B(w)$, the contribution of Type-3 forms towards the objective function is at least $2^R(\#B(w))^{p/2}$. Hence the contribution of blocks in (6.2) is at least,

$$\sum_{B(w):\#B(w)\geq k^3} 2^R(\#B(w))^{p/2}$$

which is minimized when all $\#B(w)$ are equal to k^3 and there are $\frac{m}{k^3 40\log R}$ of them. Thus the contribution is at least

$$2^R(k^3)^{p/2}\frac{m}{k^3 40\log R} \;\gg\; k^p 2^R m$$

Note that the contribution in the completeness case is $2^R m$ and therefore one gets a penalty of factor k^p as desired.

Finishing the Proof

After handling all the annoying cases, we can now assume that $\#\mathbf{x} \geq \frac{m}{20\log R}$ and that

$$\sum_{B(w):\#B(w)\geq k^3} \#B(w) \;\leq\; \frac{m}{40\log R}$$

This implies that for at least $\frac{m}{k^3 40 \log R} = \delta m$ vertices $w \in W$, the block $B(w)$ contains at least one non-zero variable. Let

$$W' = \{\, w \mid B(w) \text{ contains at least one non-zero variable}\}$$

We have $|W'| \geq \delta |W|$. Fix one non-zero variable in $B(w)$ for every $w \in W'$ and let this variable be $x_{w,A(w)}$ (thus we get an assignment A of labels to vertices in W').

By an averaging argument, for at least $\delta/4$ fraction of the vertices $v \in V$, at least $\delta/4$ fraction of their neighbors are in W'. Call any such vertex v "good". For any good vertex v, let

$$\Psi(v) = \{\, \pi^{v,w}(A(w)) \mid w \in W', w \in N(v)\}$$

Lemma 6.5.2 *For at most half the good vertices* v, $|\Psi(v)| \leq k$.

Proof: Assume on the contrary that for half of the good vertices v, $|\Psi(v)| \leq k$. Thus for every such vertex v, we can assign at most k labels such that for every neighbor $w \in W'$ of v, the label $\pi^{v,w}(A(w))$ is included. Choosing at random, one of the at most k labels for every such vertex v, gives a labeling to the Label Cover problem that satisfies following fraction of edges :

$$\left(\frac{1}{2}\frac{\delta}{4}\right)\frac{\delta}{4}\frac{1}{k} = \frac{1}{32 \cdot 1600 (\log R)^2 k^7} \gg \frac{1}{R^\gamma} \qquad \text{since } k = R^{\gamma/20}$$

This contradicts Theorem 6.2.1. ∎

Hence we can assume that for at least half of the good vertices v, $|\Psi(v)| \geq k$. For any such vertex v, assume w.l.o.g. that $\{1, 2, \ldots, k\} \subseteq \Psi(v)$. Thus for every $1 \leq j \leq k$, there exists $w_j^* \in N(v)$ such that $\pi^{v,w_j^*}(A(w_j^*)) = j$. Hence for any sequence $(w_1^*, w_2^*, \ldots, w_k^*, w_{k+1}, \ldots, w_S)$ where $w_{k+1}, \ldots, w_S \in N(v)$ are arbitrary, the

Type-4 linear form has at least k non-zero variables. Applying Lemma 6.5.1, we get a contribution of at least $2^R D^{S-k} k^{p-k}$ towards the objective function.

$\delta/4$ fraction of the vertices v are good, hence the objective function is at least

$$(\frac{\delta}{8}n)2^R D^{S-k} k^{p-k}$$

Note that the contribution in the completeness case is at most $2^R n D^S$. We will choose p such that

$$\frac{\delta}{8}n2^R D^{S-k} k^{p-k} \ > \ 2^R n D^S \cdot k^{(1-\epsilon)p}$$

Thus we get a penalty of factor $k^{(1-\epsilon)p}$ in the soundness case. Noting that $D \leq R, k = R^{\gamma/20}$, we see that it suffices to take $p = \frac{20k}{\epsilon\gamma}$. This gives hardness factor of $k^{1-\epsilon}$ or $p^{1-\epsilon'}$ as desired. This completes the full reduction.

Remark : It is crucial that the soundness parameter of the Label Cover problem is $1/R^\gamma$, i.e. polynomially small in the domain size R. Thus we use Raz's Parallel Repetition Theorem in a very strong sense, namely, the error goes down exponentially with the number of repetitions.

Chapter 7

Hardness of Vertex Cover in k-Uniform Hypergraphs

Vertex Cover in a graph is a set of vertices that touches every edge. A trivial algorithm gives a 2-approximation for minimum vertex cover and no better algorithm is known. It is a major open problem to show a matching lower bound, i.e. $2 - \epsilon$ hardness for vertex cover for every $\epsilon > 0$. Arora et al [10] show an *integrality gap* of $2 - \epsilon$ for a large family of linear programs for vertex cover, implying that LP-based approaches are unlikely to give an approximation ratio better than 2.

Hardness of vertex cover is intimately related to a fundamental (and deep) open problem about PCPs. Showing $2 - \epsilon$ hardness for vertex cover is equivalent to showing that given a graph containing an independent set of (relative) size $\alpha = \frac{1}{2} - \epsilon$, it is hard to find an independent set of size ϵ. This is in turn equivalent to constructing a PCP with completeness α, soundness ϵ and *zero free bits*, meaning the verifier *knows* the answer to every query before reading it ! However, at present we have no clue how to construct such a PCP, even for any fixed value of $\alpha < \frac{1}{2}$. Results of Håstad [60] and Dinur, Safra [32] showing $\frac{7}{6} - \epsilon$ and 1.36 hardness respectively, do not give such a PCP. Since we are stuck on the vertex cover problem, it is natural to consider its generalization to hypergraphs.

Vertex Cover on k-uniform hypergraphs has an approximation algorithm with ratio k. The main result in this chapter is an almost tight lower bound of $k - 1 - \epsilon$ (Theorem 7.1.1) for every $k \geq 3$. We show that it is NP-hard to find an independent set of size ϵ in a k-uniform hypergraph that is guaranteed to contain an independent set of size $\alpha = 1 - \frac{1}{k-1} - \epsilon$. As pointed out, such a result for graphs is open. Our result has subsequently been used by Chuzhoy et al [26] to show optimal $\Omega(\log^* n)$ hardness for Asymmetric k-Center problem.

PCP in this chapter is analyzed using purely combinatorial methods as opposed to Fourier methods employed in earlier chapters. We use the Multi-layered Label Cover problem (Theorem 4.2.4) and biased long code (Definition 7.2.2). In Chapter 8, we show that the Unique Games Conjecture in fact implies a factor $k-\epsilon$ hardness for every $k \geq 2$.

7.1 Results and Techniques

A k-uniform hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{E})$ consists of a set of vertices $\mathcal{V}$ and a collection $\mathcal{E}$ of k-element subsets of $\mathcal{V}$ called hyperedges. A *vertex cover* of $\mathcal{H}$ is a subset $S \subseteq \mathcal{V}$ such that every hyperedge in $\mathcal{E}$ intersects S, i.e. $e \cap S \neq \emptyset$ for each $e \in \mathcal{E}$. An *independent set* in $\mathcal{H}$ is a subset whose complement is a vertex cover, or in other words a subset of vertices that contains no hyperedge entirely within it. The Ek-Vertex-Cover problem is the problem of finding a minimum size vertex cover in a k-uniform hypergraph. This problem is alternatively called the minimum hitting set problem with sets of size k and it is equivalent to the set cover problem where each element of the universe occurs in exactly k sets.

A very simple approximation algorithm is the following: greedily pick a maximal set of pairwise disjoint hyperedges, take all vertices in the chosen hyperedges and declare it to be a vertex cover. It is easy to show that this gives a factor k approximation algorithm for Ek-Vertex-Cover. State of the art techniques yield only a tiny improvement, achieving a $k - o(1)$ approximation ratio [56]. This raises the question whether achieving an approximation factor of $k - \epsilon$ for any constant $\epsilon > 0$ could be NP-hard. In this chapter, we prove the following nearly tight hardness result. It appears in [29].

Theorem 7.1.1 *For every $k \geq 3$, Ek-Vertex-Cover is NP-hard to approximate within factor $k - 1 - \epsilon$ for arbitrarily small constant $\epsilon > 0$.*

Previous Hardness Results

The vertex-cover problem on hypergraphs where the size of the hyperedges is unbounded is nothing but the Set-Cover problem. For this problem there is a $\ln n$ approximation algorithm [91, 67], and a matching $(1 - o(1)) \ln n$ hardness result due to Feige [37]. The first explicit hardness result shown for Ek-Vertex-Cover was due to Trevisan [111] who considered the approximability of bounded degree instances of several combinatorial problems, and specifically showed an inapproximability factor of $k^{1/19}$ for Ek-Vertex-Cover. Holmerin [66] showed that E4-Vertex-Cover is NP-hard to approximate within $2 - \epsilon$, and more recently [65] obtained $k^{1-\epsilon}$ hardness for Ek-Vertex-Cover and $\frac{3}{2} - \epsilon$ hardness for E3-Vertex-Cover. Goldreich [51] showed a direct 'FGLSS'-type [38] reduction (involving no use of the long-code, a crucial component in most recent PCP constructions) attaining a hardness factor of $2 - \epsilon$ for Ek-Vertex-Cover for some constant k.

Quite surprisingly, Dinur, Guruswami and Khot [28] gave a fairly simple proof of $\frac{k}{3}$ hardness result for Ek-Vertex-Cover. The proof takes a combinatorial view of Holmerin's construction and instead of Fourier analysis uses some properties of intersecting families of finite sets. They also give a more complicated reduction that shows a factor $k-3-\epsilon$ hardness for Ek-Vertex-Cover. The crucial impetus for their work came from the recent result of Dinur and Safra [32] on the hardness of vertex cover (on graphs), and as in [32] the notion of biased long codes and some extremal combinatorics play an important role. In addition to ideas from [32], the factor $k-3-\epsilon$ hardness result also exploits the notion of covering complexity introduced by Guruswami, Håstad and Sudan [55].

Techniques and Overview of the Reduction

Theorem 7.1.1 is proved using a reduction from the Multi-layered Label Cover problem in Theorem 4.2.4. The construction in this chapter differs from the constructions in previous chapters in a crucial way : we take a combinatorial view of the long code and use theorems from extremal combinatorics instead of the Fourier analysis methods.

As defined in earlier chapters, a long code over domain M is indexed by all boolean functions $g : M \mapsto \{-1,1\}$. The values of the long code are ± 1 and the length of the long code is $2^{|M|}$. In the combinatorial view of the long code, we let the bits be indexed by all subsets of M and the code is $\{0,1\}$ valued. An encoding of element $b \in M$ is defined as follows :

$$\text{For } F \subseteq M, \text{ the bit indexed by } F \text{ equals } 1 \text{ if and only if } b \in F$$

As we will see, this enables us to view the tests of a PCP verifier as constraints on set-families and apply powerful machinery of extremal combinatorics. Here is the overview of our reduction. Let

$$\mathcal{L}_{multi}(G, \{W_i\}_{i=0}^{L}, E = \cup_{0 \le i < j \le L} E_{ij}, \{M_i\}_{i=0}^{L}, \{\pi^{v,w}\}_{(v,w)\in E})$$

be an instance of the multi-layered Label Cover with vertices partitioned into layers $W_0, W_1, \ldots, W_L$, and a set of edges E_{ij} between layers i and j. The goal is to build a k-uniform hypergraph from $\mathcal{L}_{multi}$ with the following properties :

1. If $\mathcal{L}_{multi}$ (or the underlying Gap-3SAT-5) is a YES instance, then the hypergraph has an independent set of size $1 - \frac{1}{k-1} - \epsilon$. The size is measured as a fractional size relative to the size of the whole hypergraph.

2. If $\mathcal{L}_{multi}$ is a NO instance, then the hypergraph has no independent set of size δ.

Here ϵ, δ can be made arbitrarily small. Vertex cover is just the complement of independent set, and therefore the size of the vertex cover is either $\le \frac{1}{k-1} + \epsilon$ or $\ge 1 - \delta$. This proves that it is NP-hard to approximate vertex cover within factor $k - 1 - \epsilon'$.

Caution : The Label Cover instance and the hypergraph built from it, both contain "vertices" and "edges". It should be clear from the context what a "vertex" refers to. The edges of the hypergraph will be called "hyperedges".

Following our standard recipe, we construct a PCP where the verifier expects as a proof the long code of the label for every vertex w of $\mathcal{L}_{multi}$. We build a hypergraph as follows : we let the bits in the proof as the vertices of the hypergraph. Let $B[w]$ be the block of vertices of the hypergraph corresponding to the bits in the long code for w. Note that if $w \in W_i$ and if M_i is the set of labels for vertices in layer i, then the block $B[w]$ contains $2^{|M_i|}$ vertices. There is one vertex for every subset of M_i (recall the combinatorial view of the long code).

For every edge $(v, w) \in E_{ij}$, we define hyperedges of the hypergraph. Every hyperedge contains exactly k vertices, one vertex from the block $B[v]$ and $k-1$ vertices from the block $B[w]$. These hyperedges are supposed to enforce the constraint given by the projection map $\pi^{v,w} : M_j \mapsto M_i$. We will later see the precise manner in which the hyperedges are defined, but here is an important thing to keep in mind :

Hyperedges are defined to ensure this property : If $\mathcal{L}_{multi}$ is a YES instance, take a correct labeling to $\mathcal{L}_{multi}$ and encode the labels with correct long codes. Then the vertices of the hypergraph that correspond to bits set to 1 form an independent set.

Thus, in the completeness case, we can identify a large independent set. Since half of the bits in a long code are 1, this is an independent set of size $\frac{1}{2}$. However, we desire an independent set of size $1 - \frac{1}{k-1} - \epsilon$ and here comes another trick. Instead of giving equal weight to all the bits in a long code, we define a *biased long code* which gives different weights to different vertices. With this trick, it can indeed be ensured that the weight of the 1-bits in a long code is $1 - \frac{1}{k-1} - \epsilon$.

In soundness case, we show that if there is an independent set of size δ, then it is possible to "decode" the long codes and define a labeling for $\mathcal{L}_{multi}$ that satisfies a good fraction of edges of $\mathcal{L}_{multi}$ giving a contradiction. Usually, this would be achieved using Fourier analysis and using the Fourier coefficients to do the (probabilistic) decoding. We however do not use Fourier analysis and the decoding relies on results from extremal combinatorics.

Motivation for using the multi-layered Label Cover : With this overall plan for the reduction, it is easy to see why we use multi-layered version of Label Cover. The hypergraph we build has $L+1$ layers, and there are hyperedges between every pair of layers. Since there are no hyperedges between vertices of the same layer, every layer is an independent set. Now think what would happen if we had only two layers. We would trivially have an independent set of size $\frac{1}{2}$ (either the left or the right layer). We however want to claim that in soundness case, there is no independent set of size δ and we are thus doomed. The multi-layered version avoids this pathological case. The *weak expansion property* (see Theorem 4.2.4) is also crucial. If the Label Cover instance had a large set of vertices with no edges among them, the hypergraph would also have a large set of vertices with no hyperedges among them (i.e. a large independent set). The weak expansion property avoids this another pathological case.

Location of the Gap

Our hardness result has the gap between sizes of the vertex cover at the "strongest" location. Specifically, to prove a factor $k-1-\epsilon'$ hardness we show that it is hard to distinguish between k-uniform hypergraphs that have a vertex cover of size $\frac{1}{k-1}+\epsilon$ from those whose minimum vertex cover has size at least $1-\delta$. This result is stronger than a gap of about $k-1$ achieved, for example, between vertex covers of size $\frac{1}{(k-1)^2}$ and $\frac{1}{k-1}$. In fact, by adding dummy vertices, our result implies that for any $c<1$ it is NP-hard to distinguish between hypergraphs that have a vertex cover of size $\frac{c}{k-1}+\epsilon$ from those whose minimum vertex-cover has size at least c. Put another way, our result shows that for k-uniform hypergraphs for $k \geq 3$, there is a fixed α such that for arbitrarily small $\delta > 0$, it is NP-hard to find an independent set of size δ even if the hypergraph is promised to contain an independent set of size α. Such a result is not known for graphs and seems out of reach of current techniques (the recent 1.36 hardness result for vertex cover on graphs due to Dinur and Safra [32], for example, shows that it is NP-hard to distinguish between cases when the graph has an independent set of size 0.38 and when there is no independent set of size 0.16).

Remark : Theorem 7.1.1 was used by Chuzhoy et al [26] to show $\Omega(\log^* n)$ hardness for Asymmetric k-Center problem. It is important for their result that our theorem gives a gap at the right location.

7.2 Biased Long Code and Intersecting Families

Definition 7.2.1 *For a bias parameter $0 < p < 1$, and a ground set M, the weight of a set $F \subseteq M$ is*

$$\mu_p^M(F) := p^{|F|} \cdot (1-p)^{|M \setminus F|}$$

When M is clear from the context, we write μ_p instead of μ_p^M . The weight of a family $\mathcal{F}$ of subsets of M (i.e. $\mathcal{F} \subseteq 2^M$) is

$$\mu_p(\mathcal{F}) := \sum_{F \in \mathcal{F}} \mu_p(F)$$

The weight of a subset is precisely the probability of obtaining this subset when one picks every element in M independently with probability p.

Definition 7.2.2 *For $0 < p < 1$, a p-biased long code over domain M is indexed by all subsets $F \subseteq M$. The bit F has a weight $\mu_p^M(F)$ attached to it. The value of the bit F in the long code of an element $b \in M$ is 1 if $b \in F$ and 0 otherwise.*

For a long code of some element $b \in M$, if $\mathcal{F}$ is the family of all sets F corresponding to 1-bits in the long code (i.e. all sets F such that $b \in F$), then $\mu_p^M(\mathcal{F}) = p$.

Let $[n] = \{0, 1, \ldots, n-1\}$ and $2^{[n]} = \{F \mid F \subseteq [n]\}$.

Definition 7.2.3 *A family $\mathcal{F} \subseteq 2^{[n]}$ is called s-wise t-intersecting if for every s sets $F_1, \ldots, F_s \in \mathcal{F}$, we have*

$$|F_1 \cap \ldots \cap F_s| \geq t$$

We are interested in bounding the size of such families, and for this purpose it is useful to introduce the notion of a left-shifted family. Performing an (i,j)-shift on a family consists of replacing the element j with the element i in all sets $F \in \mathcal{F}$ such that $j \in F$, $i \notin F$ and $(F \setminus \{j\}) \cup \{i\} \notin \mathcal{F}$. A left-shifted family is a family which is invariant with respect to (i,j)-shifts for any $1 \leq i < j \leq n$. For any family $\mathcal{F}$, by iterating the (i,j)-shift for all $1 \leq i < j \leq n$ we eventually get a left-shifted family which we denote by $S(\mathcal{F})$. The following simple lemma summarizes the properties of the left-shift operation (see [54], p. 1298, Lemma 4.2):

Lemma 7.2.4 *For any family $\mathcal{F} \subseteq 2^{[n]}$, there exists a one-to-one and onto mapping τ from $\mathcal{F}$ to $S(\mathcal{F})$ such that $|F| = |\tau(F)|$ for every $F \in \mathcal{F}$. In other words, left-shifting a family maintains its size and the size of the sets in the family. Moreover, if $\mathcal{F}$ is an s-wise t-intersecting family then so is $S(\mathcal{F})$.*

The next lemma states that a subset F in a left-shifted s-wise t-intersecting family cannot be "sparse" on all of its prefixes $F \cap [t + js]\ \forall\, j \geq 0$.

Lemma 7.2.5 ([54], p. 1311, Lemma 8.3) *Let $\mathcal{F}$ be a left-shifted s-wise t-intersecting family. Then, for every $F \in \mathcal{F}$, there exists a $j \geq 0$ with $|F \cap [t + sj]| \geq t + (s-1)j$.*

The following is the main lemma of this section. It shows that for any $p < \frac{s-1}{s}$, a family with non-negligible μ_p-weight cannot be s-wise t-intersecting for sufficiently large t.

Lemma 7.2.6 *For any δ, s, p with $p < \frac{s-1}{s}$, there exists a $t = t(\delta, s, p)$ such that for any s-wise t-intersecting family $\mathcal{F} \subseteq 2^{[n]}$, $\mu_p(\mathcal{F}) < \delta$.*

Proof: Let $\mathcal{F}$ be an s-wise t-intersecting family where t will be determined later. According to Lemma 7.2.4, $S(\mathcal{F})$ is also s-wise t-intersecting and $\mu_p(S(\mathcal{F})) = \mu_p(\mathcal{F})$. By

Lemma 7.2.5, for every $F \in S(\mathcal{F})$, there exists a $j \geq 0$ such that $|F \cap [t + sj]| \geq t + (s-1)j$. We can therefore bound $\mu_p(S(\mathcal{F}))$ from above by the probability that such a j exists for a random set chosen according to the distribution μ_p.

Let $\theta = \frac{s-1}{s} - p$. Then, for any $j \geq 0$, Pr[$|F \cap [t + sj]| \geq t + (s-1)j$] is at most

$$\text{Pr}[\ |F \cap [t + sj]| - p(t + sj) \geq \theta(t + sj)\ ] \leq e^{-2(t+sj)\theta^2}$$

using Chernoff bound. Summing over all $j \geq 0$, we get

$$\mu_p(S(\mathcal{F})) \leq \sum_{j \geq 0} e^{-2(t+sj)\theta^2} = e^{-2t\theta^2}/(1 - e^{-2s\theta^2})$$

which is smaller than δ for large enough t. ■

7.3 The Hypergraph Construction

We are now ready to prove Theorem 7.1.1. The overall plan for the proof is already outlined in Section 7.1.

Let $\mathcal{L}_{multi}(G, \{W_i\}_{i=0}^{L}, E = \bigcup_{i<j} E_{ij}, \{M_i\}_{i=0}^{L}, \{\pi^{v,w}\}_{(v,w)\in E})$ be an instance of the multi-layered Label Cover given by Theorem 4.2.4. Fix $k \geq 3$ and arbitrarily small $\epsilon, \delta > 0$. Define the bias parameter $p = 1 - \frac{1}{k-1} - \epsilon$. We present a construction of a k-uniform hypergraph $\mathcal{H}$.

Defining Vertices of Hypergraph

The set of vertices of the hypergraph $\mathcal{H}$ is defined to be

$$\{(w, F) \mid w \in W_i,\ F \subseteq M_i,\ \ 0 \leq i \leq L\}$$

For a fixed $w \in W_i$, the block of vertices $B[w]$ is defined to be

$$B[w] := \{(w, F) \mid F \subseteq M_i\}$$

The weight of the vertices inside the block $B[w]$ is according to $\mu_p^{M_i}$, i.e. the weight of a vertex $(w, F), F \subseteq M_i$ is proportional to $\mu_p^{M_i}(F) = p^{|F|}(1-p)^{|M_i \setminus F|}$ as in Definition 7.2.1. For every layer i, all blocks $B[w]$ for $w \in W_i$ have the same total weight and the total weight of each layer is $\frac{1}{L+1}$. Formally, the weight of a vertex (w, F) for $w \in W_i$ is given by

$$\frac{1}{L+1} \frac{1}{|W_i|} \mu_p^{M_i}(F)$$

Thus the total weight of all vertices in the hypergraph is equal to 1.

It is clear that the vertices in block $B[w]$ correspond to the bits in the biased long code for the label of w.

Defining Hyperedges

Let $0 \leq i < j \leq L$ be any two layers and $v \in W_i, w \in W_j$ be adjacent vertices in these layers, i.e. $(v, w) \in E_{ij}$. Let $\pi^{v,w} : M_j \mapsto M_i$ be the corresponding projection map.

We define hyperedges between vertices in blocks $B[v]$ and $B[w]$ as follows : every hyperedge has k vertices, one vertex from the block $B[v]$ and $k - 1$ vertices from the block $B[w]$. For $I \subseteq M_i$ and $F_1, F_2, \ldots, F_{k-1} \subseteq M_j$, we define

$\big((v, I),\ (w, F_1),\ (w, F_2), \ldots, (w, F_{k-1}) \big)$ to be a hyperedge if and only if

$$I \cap\ \pi^{v,w} \left(\cap_{l=1}^{k-1} F_l \right) = \emptyset \tag{7.1}$$

The hyperedges are defined to ensure that there is a large independent set in completeness case, as will be clear next.

Completeness

Assume that $\mathcal{L}_{multi}$ is a YES instance and $\Phi : W_i \mapsto M_i$ for $0 \le i \le L$ be the labeling of its vertices. We will show that if these labels are encoded using correct biased long codes, then the hypergraph vertices corresponding to 1-bits in the long codes form an independent set. Specifically let

$$IS(\mathcal{H}) := \{(w, F)\, | w \in W_i,\ F \subseteq M_i,\ \Phi(w) \in F,\ 0 \le i \le L\, \}$$

We claim that $IS(\mathcal{H})$ is an independent set in the hypergraph. Suppose on the contrary that this is not the case. Therefore, there exist k vertices in $IS(\mathcal{H})$ that form a hyperedge. Let these vertices be

$$(v, I), (w, F_1), (w, F_2), \ldots (w, F_{k-1}) \qquad \text{where } v \in W_i, w \in W_j, I \subseteq M_i, F_l \subseteq M_j$$

By definition of $IS(\mathcal{H})$, we have

$$\Phi(v) \in I, \qquad \Phi(w) \in \cap_{l=1}^{k-1} F_l$$

Also, since Φ is a correct labeling, we have $\pi^{v,w}(\Phi(w)) = \Phi(v)$ and therefore

$$\Phi(v) \in I \cap \pi^{v,w} \left(\cap_{l=1}^{k-1} F_l \right)$$

In particular, this intersection is non-empty which contradicts the way hyperedges are defined (see Equation (7.1)).

The weight of the independent set $IS(\mathcal{H})$ is $p = 1 - \frac{1}{k-1} - \epsilon$ since the weight of all the 1-bits of a p-biased long code is p. We now turn to the soundness of the construction.

7.4 Soundness of the Construction

We show that if $\mathcal{L}_{multi}$ is a NO instance, then $\mathcal{H}$ has no independent set of size δ. Suppose on the contrary that $IS(\mathcal{H})$ is an independent set of size δ. We will obtain a labeling to two layers i_0, j_0 of the Label Cover instance that satisfies a significant fraction of edges between these layers. Soundness property of Theorem 4.2.4 then gives a contradiction.

By an averaging argument, there are at least $\delta/2$ fraction of vertices w of Label Cover for which at least $\delta/2$ fraction of the vertices in block $B[w]$ are in $IS(\mathcal{H})$. Call such w's good. Thus defining

$$\mathcal{F}[w] := \{F \mid (w, F) \in IS(\mathcal{H})\}$$

we have

$$w \text{ is good} \qquad \text{if and only if} \qquad \mu_p(\mathcal{F}[w]) \geq \delta/2$$

Again, by an averaging argument, there are at least $\delta/4$ fraction of the Label Cover layers 0-through-L which contain at least $\delta/4$ fraction of good w-vertices. By the *weak expansion property* of the multi-layered Label Cover instance (Theorem 4.2.4), if $L \geq O(1/\delta^2)$, there are two layers i_0, j_0 such that (fix these layers for the rest of the proof) :

- At least $\delta/4$ fraction of vertices in layers i_0, j_0 are good. Let $Y \subseteq W_{i_0}, X \subseteq W_{j_0}$ be the sets of good vertices with $|Y| \geq \delta|W_{i_0}|/4$, $|X| \geq \delta|W_{j_0}|/4$. Also, let $N = M_{i_0}$ and $M = M_{j_0}$ be the sets of labels for layers i_0, j_0 respectively.

- The number of Label Cover edges between X and Y is at least $\delta^2/64$ fraction of the total number of edges between layers i_0, j_0.

For any $w \in X$, the set family $\mathcal{F}[w]$ is a family with μ_p-weight at least $\delta/2$. According to Lemma 7.2.6, there exists $t = t(\frac{\delta}{2}, k-1, p)$ and $k-1$ sets $F_{w,1}, F_{w,2}, \ldots, F_{w,k-1} \in \mathcal{F}[w]$ that intersect in less than t labels. Let

$$C[w] := \bigcap_{l=1}^{k-1} F_{w,l} \qquad \text{with } C[w] \subseteq M,\ |C[w]| \leq t$$

$C[w]$ will be the set of candidate labels for w.

In the following we define an assignment of labels Φ to the vertices in X and Y such that many of the Label Cover edges between them are satisfied. For $w \in X$, define its label $\Phi(w)$ to be a random label from the set $C[w]$. For $v \in Y$, we choose a label

$$\Phi(v) := \text{ maximizer}_{\,a\,\in\,N}\ \ |\{w \in X \mid a \in \pi^{v,w}(C[w])\ \}|$$

i.e., the label that is contained in the largest number of projections of $C[w]$ where w ranges over all vertices in X such that (v, w) is an edge of Label Cover.

Before continuing, we need the following simple lemma :

Lemma 7.4.1 *Let $A_1, \ldots, A_n$ be a collection of n sets of size at most m such that no element is contained in more than h sets. Then, there are at least $\frac{n}{1+(h-1)m} \geq \frac{n}{hm}$ disjoint sets in this collection.*

Proof: We prove by induction on n that there are at least $\frac{n}{1+(h-1)m}$ disjoint sets in the collection. The claim holds trivially for $n \leq 1 + (h-1)m$. Otherwise, consider all the sets that intersect A_1. Since no element is contained in more than h sets, the number of such sets (including A_1) is at most $1 + (h-1)m$. Removing these sets we get, by using the induction hypothesis, a collection that contains $\frac{n-1-(h-1)m}{1+(h-1)m} = \frac{n}{1+(h-1)m} - 1$ disjoint sets. We conclude the induction step by adding A_1 to the disjoint sets. ■

Consider a vertex $v \in Y$ and a vertex $w \in X$ such that (v, w) is an edge of Label Cover. Since $IS(\mathcal{H})$ is an independent set, there is no hyperedge of the form

$$((v, I), (w, F_{w,1}), \ldots, (w, F_{w,k-1}) \quad \text{for any} \quad I \in \mathcal{F}[v]$$

Therefore, every $I \in \mathcal{F}[v]$ must intersect $\pi^{v,w}(\cap_{l=1}^{k-1} F_{w,l}) = \pi^{v,w}(C[w])$ (recall the definition of hyperedges, Equation (7.1)). Now consider the family of projections $\pi^{v,w}(C[w])$ for all $w \in X$ such that (v, w) is an edge of Label Cover. Let q denote the maximum number of disjoint sets inside this family. Note that every disjoint set reduces the weight of the vertices in $\mathcal{F}[v]$ by a factor of $1 - (1-p)^t$. Because the weight of $\mathcal{F}[v]$ is at least $\frac{\delta}{2}$, we obtain that q is at most $\log(\frac{\delta}{2}) / \log(1 - (1-p)^t)$. Claim 7.4.1 implies that there exists a label for v that is contained in at least a fraction

$$\frac{1}{t \log(\frac{\delta}{2}) / \log(1 - (1-p)^t)}$$

of the projections $\pi^{v,w}(C[w])$. Therefore, the expected fraction of Label Cover edges satisfied between X and Y is at least

$$\frac{1}{t^2 \log(\frac{\delta}{2})/\log(1-(1-p)^t)}$$

Now at least $\delta^2/64$ fraction of edges between layers i_0, j_0 are between X and Y. Thus we get a labeling for layers i_0, j_0 satisfying a significant fraction of edges. As usual, choosing the soundness parameter of Label Cover small enough (i.e. choosing the parameter u in Theorem 4.2.4 large enough) suffices for a contradiction.

Chapter 8

Unique Games Conjecture and its Consequences

The discovery of PCP Theorem and subsequent powerful PCP constructions have led to (in many cases optimal) hardness of approximation results for various optimization problems, such as MaxClique [59], MAX-3SAT [60] and Set Cover [37]. However PCP techniques haven't been successful in obtaining *good* hardness results for some problems like Vertex Cover and Min-2SAT-Deletion. In this chapter, we try to identify some promising new directions for attacking these problems.

We propose a conjecture called Unique Games Conjecture (Conjecture 8.1.1) that states existence of an outer verifier different from all known ones. The conjecture has strong implications. It implies any constant factor hardness for Min-2SAT-Deletion (Theorem 8.3.1) and optimal $2 - \epsilon$ hardness for Vertex Cover (Theorem 9.0.2). These implications are highly non-trivial and rely on difficult Fourier analysis results, namely, Bourgain's Theorem and Friedgut's Theorem.

We emphasize that identifying verifiers with right properties has been crucial in PCP literature. For example, Bellare and Sudan [17] noted that a verifier with *low amortized free bit complexity* is sufficient to give $n^{1-\epsilon}$ hardness for MaxClique. Bellare et al [16] later showed that such a verifier is in fact necessary and then Håstad [60] was able to construct one such verifier.

We view our conjecture as more of a question, in that we don't have an intuition one way or the other. We also explore some simple ideas for proving and refuting the conjecture and describe why they don't work.

8.1 Conjecture, its Motivation and Results

All PCP constructions today (with the possible exception of [32]) follow the basic paradigm of composing a so-called "outer verifier" with an "inner verifier". Most of recent research has focused on improving the quality of the inner verifier. Many sophisticated inner verifiers have been constructed (see [59], [60], [106], [55], and previous chapters of this thesis) based on long codes and Fourier analysis techniques. However the outer verifier has remained untouched. All PCP constructions use the same outer verifier, namely the Raz Verifier (or the Label Cover problem) obtained by parallel repetition of a 2-Prover-1-Round protocol for Gap-3SAT. The soundness property of this verifier is given by Raz's Parallel Repetition Theorem.

In this chapter, we show that one promising approach to attack problems for which PCP techniques have failed so far, is to construct an outer verifier with "better properties". The Raz Verifier is a 2-Prover-1-Round game (see Section 2.2) with the following crucial properties :

1. For arbitrarily small $\delta > 0$, it is NP-hard to determine whether the value of the game is 1 or at most δ.

2. The answers of the provers are from a domain of size k where k is a constant depending on δ.

3. The answer of the second prover uniquely determines the answer of the first prover.

One might expect Property (3) to be even stronger, i.e. the answer of the second prover uniquely determines the answer of the first prover and vice versa. In fact such games have been considered in literature before ([43], [35]) and they are called "unique games". However, to the best of our knowledge, the question whether unique 2-prover games (with $(1-\zeta, \delta)$ gap in their value) are powerful enough to capture NP hasn't been considered before. This question is precisely the focus of this chapter and we make the following (rather bold) conjecture :

Conjecture 8.1.1 *(The Unique Games Conjecture :) For arbitrarily small constants $\zeta,\ \delta > 0$, there exists a constant $k = k(\zeta, \delta)$ such that it is NP-hard to determine whether a unique 2-prover game with answers from a domain of size k has value at least $1 - \zeta$ or at most δ.*

Remark : One can trivially determine whether a unique 2-prover game has value 1. Therefore the gap in the above conjecture is $(1-\zeta, \delta)$ as opposed to the gap $(1, \delta)$ in the Raz Verifier. In other words, NP-hard unique games must lose perfect completeness.

We show that a positive resolution of this conjecture would have many interesting consequences. We use the 2-prover game given by the conjecture as an outer PCP verifier and build appropriate inner verifiers to prove the following results :

1. Let Max-2-Lin-2 be a problem where we are given a system of linear equations modulo 2, each equation containing exactly 2 variables. The goal is to find an assignment that satisfies maximum number of equations.

   We show that the Unique Games Conjecture implies : For every $\frac{1}{2} < t < 1$, for all sufficiently small $\epsilon > 0$, it is NP-hard to distinguish between instances of Max-2-Lin-2 where either there exists an assignment satisfying at least $1 - \epsilon$ fraction of equations or no assignment satisfies more than $1 - \epsilon^t$ fraction of equations.

   This hardness result is tight since the algorithm of Goemans and Williamson [50] for Max-2-Lin-2, on an instance with optimum $1 - \epsilon$, produces a solution with value $1 - O(\sqrt{\epsilon})$.

2. A reduction from Max-2-Lin-2 to 2SAT gives $(1 - \epsilon, 1 - \epsilon^t)$ gap for 2SAT for any $\frac{1}{2} < t < 1$. As a corollary, it is NP-hard to approximate Min-2SAT-Deletion (also called Min-2CNF-Deletion) within any constant factor. On the algorithmic side, Zwick's algorithm [116], on a 2SAT instance with optimum $1 - \epsilon$ produces an assignment with value $1 - O(\epsilon^{1/3})$. Klein et al [80] give $O(\log n \log \log n)$ approximation for Min-2SAT-Deletion.

3. Assuming the Unique Games Conjecture, we also show that vertex cover on graphs is NP-hard to approximate within factor $2 - \epsilon$ for any constant $\epsilon > 0$. In fact, vertex cover on k-uniform hypergraphs is NP-hard to approximate within factor $k - \epsilon$ for every $k \geq 2$.

In light of such interesting consequences of the Unique Games Conjecture, we think it is an important open problem to prove or disprove it. In this chapter, we also present a semi-definite programming based algorithm that gives the following theorem :

Theorem 8.1.2 *There exists a (poly-time) algorithm such that given a unique 2-prover game with value* $1-\epsilon$ *and answers from a domain of size* k*, it finds prover strategies that make the verifier accept with probability* $1-O(k^2\epsilon^{1/5}\sqrt{\log(\frac{1}{\epsilon})})$.

Andersson et al [7] proved a similar result for the problem Max-2-Lin-p, where the constraints are linear equations mod p with every equation containing exactly 2 variables. Such constraints have the *uniqueness property* since the value to one variable in the equation uniquely determines the value to the second variable. Our algorithm is simpler and more general than that of Andersson et al.

Theorem 8.1.2 shows that if the Unique Games Conjecture is true, the domain size required $k = k(\zeta, \delta)$ must be at least $\frac{1}{\zeta^{1/10}}$. A trivial bound $k \geq \frac{1}{\delta}$ also holds, since the provers can choose their answers uniformly at random from the domain of size k and make the verifier accept with probability $\frac{1}{k}$.

Overview of the Chapter

We prove hardness of Max-2-Lin-2 and Theorem 8.1.2 in this chapter. These results and formulation of the Unique Games Conjecture appear in [77]. For proving hardness of Vertex Cover, we need to state a stronger form of the Unique Games Conjecture and show that the stronger form in fact follows from the original form. This result and hardness of Vertex Cover appear in [79] and we present them in the next chapter.

8.2 Unique Label Cover Problem

As always, we work with the Label Cover problem instead of 2-prover games. We first define a weighted version of the Label Cover problem with an additional property that the projection maps are bijections (i.e. permutations).

The Weighted Unique Label Cover Problem

Definition 8.2.1 *A weighted unique Label Cover problem $\mathcal{L}(G(V,W), M, \{\pi^{v,w}\}, \{p_{vw}\})$ consists of a complete bipartite graph $G(V,W)$, with bipartition V, W. An edge (v,w) has a weight p_{vw} with $\sum_{v,w} p_{vw} = 1$. Every vertex in $V \cup W$ is supposed to get a label from a set M. With every edge (v,w) there is associated a bijection $\pi^{v,w} : M \to M$. For an assignment Φ of labels to the vertices of the graph, that is for a function $\Phi : V \cup W \to M$, an edge (v,w) is said to be satisfied if $\pi^{v,w}(\Phi(w)) = \Phi(v)$. The goal is to find an assignment of labels that maximizes the total weight of satisfied edges. We define $OPT(\mathcal{L})$ to be the maximum weight of edges satisfied by any labeling.*

The instances of Label Cover given by Theorem 2.3.2 have $|M| \gg |N|$ and the projections $\pi^{v,w} : M \to N$ are highly many-to-one (this many-to-one-ness increases as soundness parameter decreases). The PCP constructions in this chapter need a very stringent condition that the projections be bijections. It is clear that the Unique Label Cover problem corresponds to a Unique 2-Prover Game. Hence the Unique Games Conjecture can be restated as :

Conjecture 8.2.2 *(The Unique Games Conjecture :) For arbitrarily small constants $\zeta,\ \delta > 0$, there exists a constant $k = k(\zeta, \delta)$ such that it is NP-hard to determine whether a weighted unique Label Cover instance with label sets of size k (i.e. $|M| = k$) has optimum at least $1 - \zeta$ or at most δ.*

8.3 Hardness of Max-2-Lin-2

In this section we present a proof of the following theorem.

Theorem 8.3.1 *The Unique Games Conjecture implies that for every $\frac{1}{2} < t < 1$, for all sufficiently small constants $\epsilon > 0$, it is NP-hard to distinguish between the instances of Max-2-Lin-2, where the fraction of satisfied equations is at least $1 - \epsilon$ or at most $1 - \epsilon^t$. In particular, Min-2SAT-Deletion is hard to approximate within any constant factor.*

This result is essentially due to Håstad [61]. He proposed a test for checking a long code and analyzed it using Bourgain's recent theorem [22] on Fourier spectrum of boolean functions, which itself was inspired by a question raised by Håstad. Our contribution is to introduce the Unique Games Conjecture and to show that Håstad's test can be extended to test the consistency between two long codes. This gives a PCP verifier that makes a linear test on 2 query bits, has completeness $1 - \epsilon$ and soundness $1 - \epsilon^t$.

Following the standard paradigm, the PCP verifier takes the unique Label Cover instance $\mathcal{L}$ guaranteed by Conjecture 8.2.2 and expects the proof to contain, for all vertices $v \in V$ and $w \in W$, the long codes of labels of v and w. These long codes are assumed to be folded, i.e. $A(-f) = -A(f)$ (see Definition 2.4.3 and Lemma 2.4.4).

The verifier picks an edge of Label Cover and checks that the labels along this edge satisfy the corresponding bijection. There is a technical issue of how an edge is picked. Let $p_v = \sum_w p_{vw}$. That is if an edge is picked with a probability equal to its weight, p_v is the probability that the left endpoint is v. Let $\Psi_v : W \to [0, 1]$ be defined as $\Psi_v(w) = \frac{p_{vw}}{p_v}$. That is $\Psi_v(w)$ is the conditional probability that the right endpoint of an edge is w given that the left endpoint is v.

Action of the verifier :

1. Pick $v \in V$ with probability p_v. Let A be the (supposed) long code of the (supposed) label of v.

2. Pick a random function $f : M \to \{-1, 1\}$ and a "perturbation function" $\mu : M \to \{-1, 1\}$. For each $x \in M$, $\mu(x) = 1$ with probability $1 - \epsilon$ and $\mu(x) = -1$ with probability ϵ.

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

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

   (b) (Consistency test) Pick a vertex $w \in W$ with the distribution Ψ_v. Let B be the (supposed) long code of the (supposed) label of w and $\pi = \pi^{v,w} : M \to M$ be the bijection between v and w. Accept if and only if

   $$A(f) = B(f \circ \pi)$$

   where $f \circ \pi$ denotes the composition of functions.

Remark : Håstad proposed and analyzed the codeword test. We propose the consistency test and show that Håstad's analysis can be extended to check consistency provided the Unique Games Conjecture is true.

Completeness

It is easy to see that the completeness of the test is $1 - \frac{\zeta+\epsilon}{2}$ where the outer Label Cover instance has completeness $1 - \zeta$. The test may fail due to 2 reasons : (1) The edge (v, w) picked by the verifier may be an unsatisfied edge of the Label Cover instance which happens with probability ζ. In this case, the consistency test fails. (2) In a correct proof, A is a long code of some $a \in M$. The codeword test fails when $\mu(a) = -1$ which happens with probability ϵ.

The claim about the completeness follows. Note that by the Unique Games Conjecture, ζ can be assumed to be arbitrarily small.

8.3.1 Bourgain's Theorem and Soundness Analysis

Recall that any supposed long code A has Fourier expansion

$$A = \sum_{\alpha \subseteq M} \widehat{A}_\alpha \chi_\alpha$$

A is a correct long code if there is $x \in M$ such that $\widehat{A}_{\{x\}} = 1$, i.e. its Fourier spectrum is entirely concentrated on a singleton set. A long code can be viewed as a boolean function on $\{-1,1\}^M$ that depends on exactly one coordinate. We use the following theorem of Bourgain [22]. Roughly speaking, it says that if Fourier spectrum of a boolean function is concentrated on sets of small size, then the function is essentially determined by a few coordinates. The theorem was motivated by Håstad's codeword test (i.e. $A(f) = A(f\mu)$) for checking long code.

Theorem 8.3.2 *Let A be any boolean function (for instance a supposed long code) and $k > 0$ an integer. Then for every $\frac{1}{2} < t < 1$, there exists a constant $c_t > 0$ such that*

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

We intend to show that the soundness of the PCP is $1 - \Omega(\epsilon^t)$. The probability of acceptance of the verifier is clearly

$$\text{Pr[acc]} = \frac{1}{2} E_{v,f,\mu} \left[\frac{1 + A(f)A(f\mu)}{2} + E_w \left[\frac{1 + A(f)B(f \circ \pi)}{2} \right] \right]$$

Using the Fourier expansion $A = \sum_\alpha \widehat{A}_\alpha \chi_\alpha$ we get

$$E_{f,\mu}[A(f)A(f\mu)] = E_{f,\mu}[\sum_{\alpha_1,\alpha_2} \widehat{A}_{\alpha_1} \widehat{A}_{\alpha_2} \chi_{\alpha_1}(f)\chi_{\alpha_2}(f)\chi_{\alpha_2}(\mu)]$$

The expectation over f is non-zero only if $\alpha_1 = \alpha_2 = \alpha$. Also $E_\mu[\chi_\alpha(\mu)] = (1 - 2\epsilon)^{|\alpha|}$. Hence

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

Using the Fourier expansion $B = \sum_\beta \widehat{B}_\beta \chi_\beta$, we have

$$E_{f,\mu}[A(f)B(f \circ \pi)] = E_{f,\mu}[\sum_{\alpha,\beta} \widehat{A}_\alpha \widehat{B}_\beta \chi_\alpha(f)\chi_\beta(f \circ \pi)\chi_\beta(\mu)] \tag{8.1}$$

We have

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

Substituting this in (8.1) and taking expectation over f, we see that the expectation is non-zero only if $\alpha = \pi(\beta)$. Since π is a bijection, $\beta = \pi^{-1}(\alpha)$. Thus (8.1) can be written as

$$E_{f,\mu}[A(f)B(f \circ \pi)] = \sum_\alpha \widehat{A}_\alpha \widehat{B}_{\pi^{-1}(\alpha)} (1 - 2\epsilon)^{|\alpha|}$$

Hence the probability of acceptance is

$$\begin{aligned}\Pr[\text{acc}] = &\ \tfrac{1}{2} + \tfrac{1}{4}E_v\left[\textstyle\sum_\alpha \widehat{A}_\alpha^2(1-2\epsilon)^{|\alpha|} + \sum_\alpha \widehat{A}_\alpha E_w\left[\widehat{B}_{\pi^{-1}(\alpha)}\right]\right] \\ = &\ \tfrac{1}{2} + \tfrac{1}{4}E_v[R_v + T_v]\end{aligned}$$

If this probability is $\geq 1 - \frac{1}{8}c_t\epsilon^t$ where c_t is as in Theorem 8.3.2, we have $E_v[R_v + T_v] \geq 2 - \frac{1}{2}c_t\epsilon^t$. This implies that over the choice of v, with probability at least $\frac{1}{2}$, $R_v + T_v \geq 2 - c_t\epsilon^t$. Fix any such "good" v. We have $R_v \geq 1 - c_t\epsilon^t$ and $T_v \geq 1 - c_t\epsilon^t > \frac{1}{2}$.

$$1 - c_t\epsilon^t \leq R_v \leq \sum_{\alpha\,:\,|\alpha|\,\leq\,\epsilon^{-1}} \widehat{A}_\alpha^2 + e^{-2}\sum_{\alpha\,:\,|\alpha|\,>\,\epsilon^{-1}} \widehat{A}_\alpha^2$$

$$\implies \sum_{\alpha\,:\,|\alpha|\,>\,\epsilon^{-1}} \widehat{A}_\alpha^2 \ < \ c_t\epsilon^t \tag{8.2}$$

Taking $k = \epsilon^{-1}$ in Theorem 8.3.2, we get

$$\sum_{\alpha\,:\,|\widehat{A}_\alpha|\,\leq\,\frac{1}{10}4^{-k^2}} \widehat{A}_\alpha^2 \ < \ \frac{1}{100} \tag{8.3}$$

Now we use the fact that $T_v > \frac{1}{2}$. Call α "good" if $\alpha \subseteq M$ is nonempty, $|\alpha| \leq \epsilon^{-1}$ and $|\widehat{A}_\alpha| \geq \frac{1}{10}4^{-k^2}$. We will show that the contribution of bad α's to T_v is small. First of all, since the tables are folded, $\widehat{A}_\alpha = 0$ when $|\alpha|$ is even (see Lemma 2.4.4). In particular $\widehat{A}_\alpha = 0$ when α is empty. Also

$$\left|\sum_{\alpha\,:\,|\alpha|>\epsilon^{-1}} \widehat{A}_\alpha E_w[\widehat{B}_{\pi^{-1}(\alpha)}]\right| \le \sqrt{\sum_{\alpha\,:\,|\alpha|>\epsilon^{-1}} \widehat{A}_\alpha^2}\ \sqrt{\sum_\alpha \left|E_w[\widehat{B}_{\pi^{-1}(\alpha)}]\right|^2}$$
$$\le \sqrt{\sum_{\alpha\,:\,|\alpha|>\epsilon^{-1}} \widehat{A}_\alpha^2} < \sqrt{c_t\epsilon^t}$$

where we used (8.2). Similarly we use (8.3) and show that the contribution of α's such that $|\widehat{A}_\alpha| \le \frac{1}{10}4^{-k^2}$ to T_v is at most $\frac{1}{10}$. This implies that T_v when restricted to good α's, still remains at least $\frac{1}{4}$. We have

$$\begin{aligned} E_w\left[\sum_\alpha \widehat{A}_\alpha^2 \widehat{B}_{\pi^{-1}(\alpha)}^2 \frac{1}{|\alpha|}\right] &\ge \epsilon\, E_w\left[\sum_{\alpha \text{ good}} \widehat{A}_\alpha^2 \widehat{B}_{\pi^{-1}(\alpha)}^2\right] \qquad (8.4)\\ &\ge \epsilon\, \frac{1}{100}4^{-2k^2} E_w\left[\sum_{\alpha \text{ good}} \widehat{B}_{\pi^{-1}(\alpha)}^2\right]\\ &\ge \epsilon\, \frac{1}{100}4^{-2k^2} E_w\left[\left|\sum_{\alpha \text{ good}} \widehat{A}_\alpha \widehat{B}_{\pi^{-1}(\alpha)}\right|^2\right]\\ &\ge \epsilon\, \frac{1}{100}4^{-2k^2} \left|E_w\left[\sum_{\alpha \text{ good}} \widehat{A}_\alpha \widehat{B}_{\pi^{-1}(\alpha)}\right]\right|^2\\ &\ge \epsilon\, \frac{1}{100}4^{-2k^2}\frac{1}{16} \end{aligned}$$

The expression on the second-last line is just T_v restricted to good α's which we showed to be at least $\frac{1}{4}$. Note that we are assuming that v is good itself, which holds with probability $\frac{1}{2}$.

Now we define a labeling for the Label Cover instance as follows : For a good vertex $v \in V$, pick α with probability $\widehat{A}_\alpha^2$, pick a random element of α and define it to be the label of v. For any vertex $w \in W$, pick β with probability $\widehat{B}_{\beta}^{2}$, pick a random element of β and define it to be the label of w.

It is easy to see that the weight of the edges satisfied by this labeling equals the expression (8.4). Label of v will be defined to be a random element $x \in \alpha$ and the label of w will be defined to be a random element $y \in \pi^{-1}(\alpha)$. With probability $\frac{1}{|\alpha|}$ it holds that $\pi(y) = x$ and the edge (v, w) in the Label Cover instance is satisfied.

Since the expression (8.4) is at least $\Omega(\epsilon 4^{-2k^2})$, we get a labeling that satisfies edges of total weight $\Omega(\epsilon 4^{-2k^2})$. However this contradicts the fact that $OPT(\mathcal{L}) \leq \delta$ if δ was chosen sufficiently small (see Conjecture 8.2.2). This shows that the soundness is at most $1 - \frac{1}{8}c_t\epsilon^t$ where $t > \frac{1}{2}$ is arbitrary, proving Theorem 8.3.1.

Remark : A simple gadget $(x \oplus y = 0 \mapsto \overline{x} \vee y,\ x \vee \overline{y})$ reduces Max-2-Lin-2 to 2SAT and implies a $(1-\epsilon, 1-\epsilon^t)$ gap for 2SAT for any $t > \frac{1}{2}$.

8.4 Proof of Theorem 8.1.2

In this section we prove Theorem 8.1.2. Instead of unique 2-prover games, we work in a more general setting of constraint satisfaction problems with uniqueness property.

Problem : *We are given a set X of n variables which take values from the set $[k] = \{1, 2, \ldots, k\}$. For every pair (u, v) of variables, there is a "constraint" which is a bijection $\pi^{u,v} : [k] \to [k]$. This constraint has a weight w_{uv} with $\sum_{(u,v)} w_{uv} = 1$.*

For an assignment $\mathcal{A} : X \to [k]$ to the variables, a constraint on the pair (u, v) is satisfied, if $\pi^{u,v}(\mathcal{A}(u)) = \mathcal{A}(v)$. The goal is to find an assignment that maximizes the total weight of satisfied constraints.

Algorithm : We use a semidefinite program from Feige and Lovasz's paper [43] and augment it with a suitable rounding procedure. Let us first formulate the problem as a quadratic integer program. For every variable $u \in X$, let $u_1, u_2, \dots, u_k$ be auxiliary variables taking 0-1 values. Place the following constraints :

$$u_1^2 + u_2^2 + \dots u_k^2 = 1 \quad \forall\, u \in X \tag{8.5}$$

$$u_i u_j = 0 \quad \forall\, u \in X \text{ and } \forall\, i \neq j \tag{8.6}$$

We intend that if an assignment assigns the value $i_0 \in [k]$ to a variable u, then $u_{i_0} = 1$ and $u_i = 0\ \forall i \neq i_0$. This would satisfy the constraints (8.5), (8.6). These constraints imply that for every pair (u, v) of variables

$$u_i v_j \geq 0 \quad \forall\, i, j \tag{8.7}$$

$$\sum_{1 \leq i,j \leq k} u_i v_j = 1 \tag{8.8}$$

It is easy to see that the goal is to maximize the following function subjected to the above constraints.

$$\sum_{(u,v)} w_{uv}(u_1 v_{\pi(1)} + u_2 v_{\pi(2)} + \dots u_k v_{\pi(k)}) \quad \text{where } \pi = \pi^{u,v} \tag{8.9}$$

Now we consider the semidefinite programming relaxation of the problem. We allow the variables $(u_1, \dots, u_k)$ to be vectors in a high dimensional space (in kn-dimensional space to be precise) and the constraints (8.5)-(8.8) replaced by the constraints :

$$\vec{u}_1 \cdot \vec{u}_1 + \vec{u}_2 \cdot \vec{u}_2 + \ldots + \vec{u}_k \cdot \vec{u}_k = 1 \quad \forall\, u \in X \tag{8.10}$$

$$\vec{u}_i \cdot \vec{u}_j = 0 \quad \forall\, u \in X \;\; \forall \;\; i \neq j \tag{8.11}$$

$$\vec{u}_i \cdot \vec{v}_j \geq 0 \quad \forall\, u, v \in X \;\; \forall \;\; i, j \tag{8.12}$$

$$\sum_{1 \leq i,j \leq k} \vec{u}_i \cdot \vec{v}_j = 1 \quad \forall\, u, v \in X \tag{8.13}$$

The goal is to maximize the following function subjected to the above constraints :

$$\sum_{(u,v)} w_{uv}(\vec{u}_1 \cdot \vec{v}_{\pi(1)} + \ldots + \vec{u}_k \cdot \vec{v}_{\pi(k)}) \text{ where } \pi = \pi^{u,v} \tag{8.14}$$

Observation : In any feasible solution of the SDP, for any two variables u, v, we have from the constraints (8.10), (8.11) and (8.13),

$$\|\sum_{i=1}^{k} \vec{u}_i\| = \|\sum_{j=1}^{k} \vec{v}_j\| = 1 \qquad \text{and} \qquad (\sum_{i=1}^{k} \vec{u}_i) \cdot (\sum_{j=1}^{k} \vec{v}_j) = 1$$

This implies that $\sum_{i=1}^{k} \vec{u}_i = \sum_{j=1}^{k} \vec{v}_j$. We denote $\vec{s} = \sum_{i=1}^{k} \vec{u}_i$ which is the same for all variables u and $\|\vec{s}\| = 1$.

We solve the semidefinite program and construct an assignment using the following rounding procedure.

- Choose a vector $\vec{r}$ from the normal distribution, i.e. choose every coordinate of $\vec{r}$ from the distribution $N(0, 1)$ independently.
- By replacing $\vec{r}$ by $-\vec{r}$ if needed, assume that $\vec{r} \cdot \vec{s} \geq 0$.

- Construct the following assignment $\mathcal{A}$: for every variable u, let

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

We prove the following theorem which is sufficient to prove Theorem 8.1.2.

Theorem 8.4.1 *If there exists an assignment that satisfies constraints with total weight $1-\epsilon$, then the above algorithm produces an assignment that satisfies constraints with expected weight* $1 - O(k^2 \epsilon^{1/5} \sqrt{\log(\frac{1}{\epsilon})})$.

Proof: Let $\alpha_{uv} = \sum_{1 \leq i \leq k} \vec{u}_i \cdot \vec{v}_{\pi(i)}$, $\pi = \pi^{u,v}$ which is the part of the SDP objective function (8.14) corresponding to the constraint on (u, v). By the hypothesis, the SDP has a solution with value at least $1-\epsilon$ implying that there exist vectors $(\vec{u}_i)_{u \in X, i \in [k]}$ satisfying

$$\sum_{(u,v)} w_{uv} \alpha_{uv} \geq 1 - \epsilon$$

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

Fix any (u, v) with $\alpha_{uv} \geq 1 - \frac{1}{2}\epsilon^{4/5}$. We will show that we have $\pi^{u,v}(\mathcal{A}(u)) = \mathcal{A}(v)$ with probability $1 - O(k^2 \epsilon^{1/5} \sqrt{\log(\frac{1}{\epsilon})})$. Let $\pi = \pi^{u,v}$ for simplicity. The intuition behind the proof is simple. if $\alpha_{uv} = 1$, the SDP constraints (8.10-8.13) imply that $\vec{u}_i = \vec{v}_{\pi(i)}$ $\forall\, i \in [k]$. Thus for any vector $\vec{r}$, if $\vec{r} \cdot \vec{u}_i$ is maximized for index i_0, then $\vec{r} \cdot \vec{v}_j$ is maximized at index $\pi(i_0)$. Hence the rounding procedure will assign $\mathcal{A}(u) = i_0,$ and $\mathcal{A}(v) = \pi(i_0)$, satisfying the constraint.

We however have $\alpha_{uv} \geq 1 - \frac{1}{2}\epsilon^{4/5}$ and it takes some effort to translate the intuition into a rigorous proof. We proceed to prove several simple lemmas.

Lemma 8.4.2 $\|\vec{u}_i - \vec{v}_{\pi(i)}\| \leq \epsilon^{2/5} \quad \forall\, i \in [k]$.

Proof:

$$1 - \frac{1}{2}\epsilon^{4/5} \leq \sum_i \vec{u}_i \cdot \vec{v}_{\pi(i)} \leq \sum_i \|\vec{u}_i\|\|\vec{v}_{\pi(i)}\|$$

$$\leq \sum_i \frac{\|\vec{u}_i\|^2 + \|\vec{v}_{\pi(i)}\|^2}{2} = 1$$

$$\implies \frac{\|\vec{u}_i\|^2 + \|\vec{v}_{\pi(i)}\|^2}{2} - \vec{u}_i \cdot \vec{v}_{\pi(i)} \leq \frac{1}{2}\epsilon^{4/5} \quad \forall\, i$$

$$\implies \|\vec{u}_i - \vec{v}_{\pi(i)}\|^2 \leq \epsilon^{4/5} \quad \forall\, i$$

■

Lemma 8.4.3 *If Y is distributed as $N(0,1)$,*

$$\Pr\left[|Y| > \gamma\right] \leq e^{\frac{-\gamma^2}{2}}$$

Proof: Standard inequality. ■

Lemma 8.4.4 *With probability $1 - O(k^2\epsilon^{1/5}\sqrt{\log(\frac{1}{\epsilon})})$, components of $\vec{r}$ along the directions of vectors*

$$\{\vec{u}_i\}_{i\in[k]}, \{\vec{u}_i - \vec{u}_j\}_{i\neq j},\ \{\vec{u}_i - \vec{v}_{\pi(i)}\}_{i\in[k]}$$

have magnitude in the range

$$\left[\ \epsilon^{1/5}\sqrt{\log(\frac{1}{\epsilon})},\ \sqrt{\log(\frac{1}{\epsilon})}\ \right]$$

Proof: This follows from the fact that $\vec{r}$ is distributed in a spherically symmetric manner and hence its component along any direction is distributed as $N(0,1)$. Hence for any unit vector $\vec{t}$,

$$\Pr\left[|\vec{r}\cdot\vec{t}| < \epsilon^{1/5}\sqrt{\log(\frac{1}{\epsilon})}\right] \;<\; 2\epsilon^{1/5}\sqrt{\log(\frac{1}{\epsilon})}$$

$$\Pr\left[|\vec{r}\cdot\vec{t}| > \sqrt{\log(\frac{1}{\epsilon})}\right] \;<\; \sqrt{\epsilon}$$

where the first inequality is trivial and the second follows from Lemma 8.4.3. Now we take a union bound along the $O(k^2)$ directions specified in the statement of this lemma. ■

Lemma 8.4.5 *With probability* $1 - 10k\epsilon^{1/5}\sqrt{\log(\frac{1}{\epsilon})}$*, the component of* $\vec{r}$ *along* $\vec{s}$*, that is* $|\vec{r}\cdot\vec{s}|$*, is at least* $5k\epsilon^{1/5}\sqrt{\log(\frac{1}{\epsilon})}$.

Proof: Trivial. ■

Thus except with probability $1 - O(k^2\epsilon^{1/5}\sqrt{\log(\frac{1}{\epsilon})})$, we can assume that $\vec{r}$ satisfies hypothesis of Lemma 8.4.4 and Lemma 8.4.5. Under this assumption, we prove the following three lemmas. Let $i_0 \in [k]$ be such that $\quad \vec{r}\cdot\vec{u}_{i_0} = \max_{1\le i\le k}\vec{r}\cdot\vec{u}_i$.

Lemma 8.4.6 $\|\vec{u}_{i_0}\| \ge 5\epsilon^{1/5}$.

Proof: $(\sum_{i=1}^{k}\vec{u}_i)\cdot\vec{r} = \vec{s}\cdot\vec{r} \ge 5k\epsilon^{1/5}\sqrt{\log(\frac{1}{\epsilon})}$ by Lemma 8.4.5 and i_0 is the index that maximizes $\vec{r}\cdot\vec{u}_i$. Hence $\vec{r}\cdot\vec{u}_{i_0} \ge 5\epsilon^{1/5}\sqrt{\log(\frac{1}{\epsilon})}$. But by Lemma 8.4.4, the component of $\vec{r}$ along $\vec{u}_{i_0}$ has magnitude at most $\sqrt{\log(\frac{1}{\epsilon})}$. This implies that $\|\vec{u}_{i_0}\| \ge 5\epsilon^{1/5}$. ■

Lemma 8.4.7 $\forall\, j \ne i_0,\ \ \vec{r}\cdot\vec{u}_j \le \vec{r}\cdot\vec{u}_{i_0} - 5\epsilon^{2/5}\sqrt{\log(\frac{1}{\epsilon})}$

Proof:

$$\begin{aligned}
\vec{r}\cdot\vec{u}_{i_0} - \vec{r}\cdot\vec{u}_j &= |\vec{r}\cdot\vec{u}_{i_0} - \vec{r}\cdot\vec{u}_j| \\
&= |\vec{r}\cdot(\vec{u}_{i_0} - \vec{u}_j)| \\
&\geq \|\vec{u}_{i_0} - \vec{u}_j\| \;\; \epsilon^{1/5}\sqrt{\log(\tfrac{1}{\epsilon})} \quad \text{by Lemma 8.4.4} \\
&\geq \|\vec{u}_{i_0}\| \; \epsilon^{1/5}\sqrt{\log(\tfrac{1}{\epsilon})} \qquad \text{Since } \vec{u}_{i_0} \perp \vec{u}_j \\
&\geq 5\epsilon^{2/5}\sqrt{\log(\tfrac{1}{\epsilon})} \qquad \text{by Lemma 8.4.6}
\end{aligned}$$

■

Lemma 8.4.8 $\forall\, i,\;\; |\vec{r}\cdot\vec{u}_i - \vec{r}\cdot\vec{v}_{\pi(i)}| \leq \epsilon^{2/5}\sqrt{\log(\tfrac{1}{\epsilon})}$

Proof:

$$\begin{aligned}
|\vec{r}\cdot\vec{u}_i - \vec{r}\cdot\vec{v}_{\pi(i)}| &= |\vec{r}\cdot(\vec{u}_i - \vec{v}_{\pi(i)})| \\
&\leq \sqrt{\log(\tfrac{1}{\epsilon})}\, \|\vec{u}_i - \vec{v}_{\pi(i)}\| \;\; \text{by Lemma 8.4.4} \\
&\leq \sqrt{\log(\tfrac{1}{\epsilon})}\epsilon^{2/5} \qquad \text{by Lemma 8.4.2}
\end{aligned}$$

■

Now we will show that

$$\vec{r}\cdot\vec{v}_{\pi(i_0)} = \max_{1\leq j\leq k}(\vec{r}\cdot\vec{v}_j) \tag{8.15}$$

This would imply that the assignment $\mathcal{A}$ given by the rounding procedure assigns $\mathcal{A}(u) = i_0,\ \mathcal{A}(v) = \pi(i_0)$ and the constraint on the pair (u, v) is satisfied.

Let $j \neq i_0$ be any index. By Lemma 8.4.8 and Lemma 8.4.7,

$$\vec{r}\cdot\vec{v}_{\pi(j)} \leq \vec{r}\cdot\vec{u}_j + \epsilon^{2/5}\sqrt{\log(\frac{1}{\epsilon})} \leq \vec{r}\cdot\vec{u}_{i_0} - 4\epsilon^{2/5}\sqrt{\log(\frac{1}{\epsilon})}$$

Also by Lemma (8.4.8) we have

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

It follows that

$$\vec{r} \cdot \vec{v}_{\pi(i_0)} > \vec{r} \cdot \vec{v}_{\pi(j)} \quad \forall\, j \neq i_0$$

finishing the proof of (8.15) and Theorem 8.4.1. ■

8.5 Conclusion

It seems quite difficult to prove (or disprove) the Unique Games Conjecture. Proving the conjecture is equivalent to constructing a PCP that reads 2 symbols and accepts if and only if these symbols satisfy a bijective constraint. However the current tools appear quite weak for constructing PCPs that read 2 symbols. Parallel repetition of a unique game is a unique game and one might hope to amplify the soundness by parallel repetition. However we do not have a hard instance of a unique game to begin with. Theorem 8.1.2 shows that if the Unique Games Conjecture is true, the domain size $k(\zeta, \delta) \geq \frac{1}{\zeta^{1/10}}$, thus the domain size would play a crucial role. On the other hand, disproving the conjecture may require an algorithm that gives a theorem similar to Theorem 8.1.2 and whose performance is independent of the domain size k.

A less ambitious goal (than proving the Unique Games Conjecture) would be to show that the value of a unique 2-prover game with domain size k is hard to approximate within factor $f(k)$ where $f(k) \to \infty$ as $k \to \infty$. The only known results are constant factor hardness for Max-2-Lin-2 by Håstad [60] and for Max-2-Lin-p by Andersson et al [7].

One can also consider the following relaxation of the uniqueness property. We say that a 2-prover game has "d-to-1 property" if the answer of the second prover uniquely determines the answer of the first prover and for every answer of the first prover, there are at most d answers for the second prover for which the verifier would accept. We assume d to be a fixed integer and $d \geq 2$. Consider the following conjecture :

Conjecture 8.5.1 *(d-to-1 Conjecture :) For arbitrarily small constant $\delta > 0$, there exists a constant $k = k(\delta)$ such that it is NP-hard to determine whether a 2-prover game with d-to-1 property and answers from a domain of size k has value 1 or at most δ.*

Note that in contrast with the Unique Games Conjecture, we can hope for perfect completeness in the d-to-1 Conjecture ($d \geq 2$). One can use some of the techniques from Dinur and Safra's paper [32] to show that the d-to-1 Conjecture implies the following results (we omit the proofs from this thesis) :

1. For arbitrarily small $\epsilon,\ \delta > 0$, it is hard to find an independent set of size δn in a graph which is guaranteed to have an independent set of size $(1 - \frac{1}{2^{1/d}} - \epsilon)n$. This implies a PCP with zero free bits, completeness $\Omega(1)$ and arbitrarily low soundness. The best known algorithm (see [6]), given a graph containing an independent set of linear size, finds an independent set of only sublinear size.

2. From the above result, it follows that if 2-to-1 Conjecture is true, it would imply $\sqrt{2} - \epsilon$ hardness for Vertex Cover which is better than the factor 1.36 by Dinur and Safra. In fact, Dinur and Safra do use an analog of 2-to-1 property.

Chapter 9

Hardness of Vertex Cover Based on Unique Games Conjecture

One of the major open problems in the field of inapproximability is whether vertex cover is hard to approximate within factor $2-\epsilon$ for every $\epsilon > 0$. This is known to be equivalent to a fundamental question about PCPs with *zero free bits*. The best known hardness is 1.36 due to Dinur and Safra [32] and the current techniques seem inadequate to prove $2-\epsilon$ hardness result. An integrality gap of $2-\epsilon$ is known for a large class of linear programs (see [10]).

In this chapter, we take a step towards resolving this question. We show $2-\epsilon$ hardness for vertex cover assuming the Unique Games Conjecture presented in Chapter 8. We show the following result that appears in [79].

Theorem 9.0.2 *Assuming the Unique Games Conjecture, vertex cover in k-uniform hypergraphs is NP-hard to approximate within factor $k-\epsilon$ for every $k \geq 2$ and $\epsilon > 0$ is an arbitrarily small constant.*

An algorithm with factor k approximation is known for vertex cover in k-uniform hypergraphs and hence this result is optimal. In Chapter 7, we showed factor $k - 1 - \epsilon$ hardness for this problem.

An important contribution of our work is to define a stronger form of the Unique Games Conjecture and show that it actually follows from the original form. The stronger form could be useful to obtain further consequences of the Unique Games Conjecture.

9.1 Techniques

The reduction in this chapter can be divided into two parts. In the first part, we state a stronger form of the Unique Games Conjecture and show that it is actually equivalent to the original form (Theorem 9.2.2). Recall that the Unique Games Conjecture states that it is NP-hard to distinguish whether a unique 2-prover game has value close to 1 or arbitrarily small. The stronger form states that the NP-hardness holds even with the following stronger requirement in completeness case : we require that the provers have a strategy such that over the choice of the question to the first prover, with probability close to 1, the verifier accepts for *every* question to the second prover.

We believe that this stronger form would be useful in our understanding of the unique games conjecture and an eventual resolution of this conjecture.

The second part of our reduction combines this conjecture with the combinatorial methods presented in Chapter 7. The reduction is roughly the same, constructing a k-uniform hypergraph from (strong unique) 2-prover game. We expect as a proof long codes of prover's answers. We let the bits in long code to be vertices of the hypergraph and define hyperedges to enforce consistency between provers' answers (or labels to Label Cover instance). The uniqueness property of the 2-prover game is crucial in our construction. We show that the hypergraph either has an independent set of size $1-\frac{1}{k}-\epsilon$ or has no independent set of size δ where ϵ, δ are arbitrarily small.

We employ several tools from Dinur and Safra's paper [32] including biased long code, Friedgut's Theorem on sensitivity of boolean functions and theorems in extremal set theory (see Section 9.3).

Motivation for Friedgut' Theorem : Recall that in the combinatorial view of the long code, the code is indexed by all sets $F \subseteq M$. A long code of $b \in M$ is defined by setting the bit F to 1 if and only if $b \in F$. Let us build a graph G as follows : its vertices are all sets $F \subseteq M$ and (F_1, F_2) is an edge if and only if $F_1 \cap F_2 = \emptyset$.

Note that an independent set in G corresponds to a family of pairwise intersecting sets. If we take a long code of $b \in M$ and take the family of all sets containing b, it is an independent set of size $\frac{1}{2}$ (of size p for a p-biased long code). This is a large independent set that is completely determined by *one* element, namely b. Friedgut's Theorem is a strong converse to this fact. It says that any maximal independent set (which is necessarily a monotone family) with significant size is essentially determined by a few elements. Thus any independent set of significant size can be "decoded" to give a small number of elements that determine it. We use this theorem in soundness analysis and show that if the graph contains an independent set of size δ, then one can take the corresponding decoded elements as candidate labels to Label Cover instance.

9.2 Constructing the Strong Label Cover

We state a stronger form of the Unique Games Conjecture (Conjecture 8.2.2) and show that it is actually equivalent to the original form.

Notation

Recall the definition of weighted unique Label Cover instance (Definition 8.2.1). The instance is given as $\mathcal{L}(G(V,W), M, \{\pi^{v,w}\}, \{p_{vw}\})$. We need some extra notation.

- Let $p(\mathcal{L}) = \sum_{v,w} p_{vw}$ and $p(\mathcal{L})$ is not necessarily required to be 1. $OPT(\mathcal{L})$ is defined as the maximum weight that can be satisfied by any labeling Φ.
- For $v \in V$, let $p(\mathcal{L}, v) = \sum_{w \in W} p_{vw}$.
- For an assignment of labels Φ, let $p_\Phi(\mathcal{L})$ be the weight of edges satisfied by labeling Φ. For $v \in V$, let $p_\Phi(\mathcal{L}, v)$ be the weight of edges incident on v satisfied by Φ.

The Unique Games Conjecture can be restated as :

Conjecture 9.2.1 *(Unique Games Conjecture :) For any $\zeta, \gamma > 0$, there exists a constant $|M|$ such that the following is NP-hard. Given a Unique-LC $\mathcal{L}$ with label set M and $p(\mathcal{L}) = 1$, distinguish between the case where there exists a labeling Φ such that $p_\Phi(\mathcal{L}) \geq 1 - \zeta$ and the case where for any labeling Φ, $p_\Phi(\mathcal{L}) \leq \gamma$.*

Stronger Form

A Strong Label Cover (Strong-LC) $\mathcal{L} = (G(V,W,E), \{\pi^{v,w}\})$ is defined as follows. We are given a bipartite graph $G(V,W,E)$ possibly with parallel edges in which all left degrees are equal to some constant d. In addition, there exists a bijection $\pi^{v,w}$ for each edge $(v,w) \in E$. A labeling and the edges satisfied by it are defined as before. In this section we prove the following theorem:

Theorem 9.2.2 *Assuming Conjecture 9.2.1, for any $\zeta, \gamma > 0$, there exist constants $|M|, d$ such that the following is NP-hard. Given a Strong-LC with label set M and left degrees d, distinguish between these cases :*

- *There exists a labeling in which for at least* $1-\zeta$ *fraction of vertices* $v \in V$*,* every *edge incident on* V *is satisfied.*

- *No labeling satisfies more than* γ *fraction of the edges.*

We begin with two lemmas. The first modifies the Unique-LC so that all $v \in V$ have the same weight. The second lemma rounds the edge weights. The proofs are very technical and they could be safely skipped.

Lemma 9.2.3 *Assuming Conjecture 9.2.1, for any* $\zeta, \gamma, \beta > 0$*, there exists a constant* $|M|$ *such that the following is NP-hard. Given a Unique-LC* $\mathcal{L}$ *with label set* M *and the property that* $\forall v \in V,\ p(\mathcal{L}, v) = 1$*, distinguish between the case where there exists a labeling* Φ *such that for* $1-\beta$ *fraction of* $v \in V$*,* $p_\Phi(\mathcal{L}, v) > 1-\zeta$ *and the case where for any labeling* Φ *at most* β *fraction of* $v \in V$ *have* $p_\Phi(\mathcal{L}, v) > \gamma$*.*

Proof: Consider a Unique-LC $\mathcal{L}' = (G'(V', W'), M', \{\pi'^{v,w}\}, \{p'_{vw}\})$ as given by Conjecture 9.2.1 with parameters ζ', γ' which will be chosen later. Also, let l be a large enough constant. The Unique-LC $\mathcal{L} = (G(V, W, E), \{\pi^{v,w}\})$ is defined as follows. Let $W = W'$. The set V includes $k(v)$ copies of each $v \in V'$, $v^{(1)}, \ldots, v^{(k(v))}$ where $k(v)$ is defined as $\lfloor l \cdot |V'| \cdot p(\mathcal{L}', v) \rfloor$. For every $v \in V'$, $w \in W$ and $i \in [k(v)]$ we define $\pi^{v^{(i)},w}$ as $\pi'^{v,w}$ and the weight $p_{v^{(i)},w}$ as $p'_{vw}/p(\mathcal{L}', v)$. Notice that $p(\mathcal{L}, v) = 1$ for all $v \in V$ and that $(l-1)|V'| \leq |V| \leq l|V'|$.

We first prove the completeness part. Given a labeling Φ' to $\mathcal{L}'$ that satisfies edges of weight at least $1-\zeta'$, consider the labeling Φ defined as $\Phi(v^{(i)}) = \Phi'(v)$. The weight of edges satisfied by Φ is:

$$\sum_{v\in V} p_\Phi(\mathcal{L},v) = \sum_{v\in V'} k(v)\cdot p_{\Phi'}(\mathcal{L}',v)/p(\mathcal{L}',v) \geq$$
$$\sum_{v\in V'} l|V'|\cdot p_{\Phi'}(\mathcal{L}',v) - \sum_{v\in V'} p_{\Phi'}(\mathcal{L}',v)/p(\mathcal{L}',v) \geq$$
$$l|V'|(1-\zeta') - |V'| = l|V'|(1-\zeta'-\frac{1}{l}) \geq l|V'|(1-2\zeta') \geq |V|(1-2\zeta')$$

for large enough l. This implies that for at least $1-\sqrt{2\zeta'}$ of vertices in V, $p_\Phi(\mathcal{L},v) \geq 1-\sqrt{2\zeta'}$. Hence, by choosing a small enough ζ', we get that at least $1-\beta$ of the vertices $v\in V$ satisfy $p_\Phi(\mathcal{L},v)\geq 1-\zeta$.

We now prove the soundness part. Assume we are given a labeling Φ to $\mathcal{L}$ for which at least β fraction of $v\in V$ have $p_\Phi(\mathcal{L},v)>\gamma$. Without loss of generality we can assume that for every $v\in V'$, the labeling $\Phi(v^{(i)})$ is the same for all i. That is because the constraints between $v^{(i)}$ and vertices in W are the same for all $i\in[k(v)]$. We define the labeling Φ' as $\Phi'(v)=\Phi(v^{(1)})$. The weight of edges satisfied by Φ' is:

$$\sum_{v\in V'} p_{\Phi'}(\mathcal{L}',v) \geq \frac{1}{l|V'|}\sum_{v\in V'} k(v)\cdot p_{\Phi'}(\mathcal{L}',v)/p(\mathcal{L}',v) =$$
$$\frac{1}{l|V'|}\sum_{v\in V} p_\Phi(\mathcal{L},v) \geq \frac{1}{l|V'|}\beta|V|\gamma \geq \frac{l-1}{l}\beta\gamma > \gamma',$$

for small enough γ'. ∎

Lemma 9.2.4 *Assuming Conjecture 9.2.1, for any $\zeta,\gamma,\beta>0$, there exists a constant $|M|$ such that the following is NP-hard. We are given a Unique-LC $\mathcal{L}$ with label set M and the properties that $\forall v\in V$, $p(\mathcal{L},v)=1$ and that there exists an integer $\alpha=O(|W|)$ such that the weight p_{vw} is multiple of $\frac{1}{\alpha}$ for all $v\in V$, $w\in W$. The goal is to distinguish between the case where there exists a labeling Φ such that $1-\beta$ of the V-vertices have*

$p_\Phi(\mathcal{L}, v) > 1 - \zeta$ *and the case where for any labeling* Φ *at most* β *of the* V*-vertices have* $p_\Phi(\mathcal{L}, v) > \gamma$.

Proof: Let $\mathcal{L}' = (G'(V', W'), \{\pi'^{v,w}\}, \{p'_{vw}\})$ be a Unique-LC as in Lemma 9.2.3 with parameters $\frac{\zeta}{2}, \frac{\gamma}{2}, \beta$. Let l be a large enough integer and define $\alpha = l|W|$. The Unique-LC $\mathcal{L} = (G(V, W), \{\pi^{v,w}\} \{p_{vw}\})$ has $V = V'$, $W = W'$, $\pi^{v,w} = \pi'^{v,w}$ and the weights $\{p_{vw}\}$ are defined as follows. Fix an arbitrary $w_0 \in W$. For every $v \in V$ we would like to round the weights p_{vw} down to the nearest multiple of $\frac{1}{\alpha}$. In order to maintain the property $p(\mathcal{L}, v) = 1$, we slightly increase p_{vw_0}. More formally, for every $v \in V$ and $w \neq w_0$ the weight p_{vw} is defined as $\lfloor \alpha p'_{vw} \rfloor / \alpha$. Also, for every $v \in V$, p_{vw_0} in defined as $1 - \sum_{w \in W \setminus \{w_0\}} p_{vw}$. Notice that for $w \neq w_0$, $p_{vw} \le p'_{vw}$ and that $p_{vw_0} \le p'_{vw_0} + |W|/\alpha = p'_{vw_0} + \frac{1}{l}$.

Assume that there exists a labeling Φ to $\mathcal{L}'$ in which $1 - \beta$ of the V'-vertices have $p_\Phi(\mathcal{L}', v) > 1 - \frac{\zeta}{2}$. Then, in $\mathcal{L}$, the same labeling satisfies that for $1 - \beta$ of the V-vertices $p_\Phi(\mathcal{L}, v) > 1 - \frac{\zeta}{2} - |W| \cdot \frac{1}{l|W|} > 1 - \zeta$ for large enough l. Also, a labeling Φ to $\mathcal{L}$ in which β of the V-vertices have $p_\Phi(\mathcal{L}, v) > \gamma$ satisfies that β of the V'-vertices have $p_\Phi(\mathcal{L}', v) > \gamma - \frac{1}{l} > \frac{\gamma}{2}$ for large enough l. ■

Proof: [of Theorem 9.2.2] Let $\mathcal{L}' = (G'(V', W'), \{\pi'^{v,w}\}, \{p'_{vw}\})$ be a Unique-LC as in Lemma 9.2.4 with parameters ζ', γ', β' which will be chosen later. We define the Strong-LC $\mathcal{L} = (G(V, W, E), \{\pi^{v,w}\})$ as follows. The set of vertices W equals W'. Let d be an integer that will be determined later. For each $v \in V'$ and each sequence $(w_1, \ldots, w_d)$ of W-vertices we create $\Pi_{i=1}^{d} \alpha p'_{vw_i}$ new vertices in V (notice that this number is integral). Each of these vertices is connected to $w_1, \ldots, w_d$ with the maps $\pi'^{v,w_1}, \ldots, \pi'^{v,w_d}$. The total number of vertices created from each $v \in V'$ is

$$\sum_{(w_1,\dots,w_d)\in W^d} \Pi_{i=1}^{d} \alpha p'_{vw_i} = \alpha^d (\sum_{w\in W} p'_{v,w})^d = \alpha^d$$

since $p(\mathcal{L}', v) = 1$. Hence, $|V| = \alpha^d |V'|$. Also note that $\mathcal{L}$ might contain parallel edges.

We first prove the completeness part. Assume that Φ' is a labeling to $\mathcal{L}'$ such that $1-\beta'$ of the V'-vertices have $p_{\Phi'}(\mathcal{L}', v) > 1-\zeta'$. Let Φ be the labeling to $\mathcal{L}$ assigning to each of the vertices created from $v \in V'$ the value $\Phi'(v)$ and for each $w \in W$ the value $\Phi'(w)$. Consider a vertex $v \in V'$ such that $p_{\Phi'}(\mathcal{L}', v) > 1-\zeta'$ and let W_v denote the set of vertices $w \in W$ such that the edge (v, w) is satisfied. Then the number of vertices in V that are connected only to vertices in W_v is

$$\sum_{(w_1,\dots,w_d)\in (W_v)^d} \Pi_{i=1}^{d} \alpha p'_{vw_i} = \alpha^d (\sum_{w\in W_v} p'_{vw})^d \geq \alpha^d (1-\zeta')^d$$

Therefore, the total number of vertices in V all of whose incident edges are satisfied by Φ is at least

$$\alpha^d (1-\zeta')^d (1-\beta')|V'| = (1-\zeta')^d (1-\beta')|V| > (1-\zeta)|V|$$

for small enough ζ' and β'.

We now prove the soundness part. Assume that no labeling Φ' to $\mathcal{L}'$ has more than β' of the V'-vertices with $p_{\Phi'}(\mathcal{L}', v) > \gamma'$. Let Φ be a labeling to $\mathcal{L}$ and define Φ' as follows. For each $w \in W$ let $\Phi'(w) = \Phi(w)$. For $v \in V'$ define $\Phi'(v)$ as the label in M that maximizes $p_{\Phi'}(\mathcal{L}', v)$. For $v \in V'$ and $i \in M$, let $S_{v,i} = \sum p'_{vw}$ where the sum is taken over all $w \in W$ such that $\pi'^{v,w}(\Phi'(w)) = i$. Then notice that $p_{\Phi'}(\mathcal{L}', v) = \max_i S_{v,i}$. Hence, for at least $1-\beta'$ of the V'-vertices, $S_{v,i} < \gamma'$ for all $i \in M$. Fix a vertex $v \in V'$

for which $S_{v,i} < \gamma'$ for all $i \in M$. Consider the subset $Z_v \subseteq W^d$ of tuples $(w_1, \dots, w_d)$ such that for all $i \in M$ there exists at most one j for which $\pi'^{v,w_j}(\Phi'(w_j)) = i$. A vertex in V created from such a tuple will have at most one satisfied edge. The number of such vertices created from v as above is

$$\sum_{(w_1,\dots,w_d)\in Z_v} \Pi_{i=1}^d \alpha p'_{vw_i} = \alpha^d \sum_{(w_1,\dots,w_d)\in Z_v} \Pi_{i=1}^d p'_{vw_i}$$

Notice that since $\sum_{w\in W} p'_{vw} = 1$, this defines a probability measure on W. Also note that the sum above is exactly the probability that a tuple $(w_1, \dots, w_d)$ is in Z_v where each element of the tuple is chosen according to this probability. Hence, the sum is at least $1 \cdot (1-\gamma') \cdot (1-2\gamma') \dots (1-(d-1)\gamma') \geq (1-d\gamma')^d > 1 - \frac{\gamma}{2}$ for small enough γ'. The number of satisfied edges in Φ is therefore at most

$$\begin{aligned} &\alpha^d \cdot \beta' \cdot |V'| \cdot d + \alpha^d(1-\beta')|V'|[(1-\frac{\gamma}{2}) \cdot 1 + \frac{\gamma}{2} \cdot d] = \\ &|V|(\beta' d + (1-\beta')(1-\frac{\gamma}{2}) + (1-\beta')\frac{\gamma}{2}d) < \gamma|V|d = \gamma|E| \end{aligned}$$

for a small enough β' and a large enough d. ∎

9.3 Tools from Sensitivity Analysis of Boolean Functions

We restate definitions from Section 7.2. For a universe M, let 2^M denote its power set, i.e. the family of all subsets of M. For a *bias parameter* $0 < p < 1$, the weight $\mu_p^M(F)$ of a set F is defined as

$$\mu_p^M(F) := p^{|F|}(1-p)^{|M\setminus F|}$$

We will omit the superscript M when it is clear which universe we are talking about. The weight of a family $\mathcal{F} \subseteq 2^M$ is defined as

$$\mu_p(\mathcal{F}) := \sum_{F \in \mathcal{F}} \mu_p(F).$$

Note that the bias parameter defines a distribution on 2^M, where a subset is picked by independently picking every element in M with probability p. We denote this distribution by μ_p^M.

9.3.1 Friedgut's 'Core' Theorem

For a family $\mathcal{F} \subseteq 2^M$, an element $\sigma \in M$ and a bias parameter p, we define the *influence of the element on the family* as

$$\mathrm{Influence}_p^M(\mathcal{F}, \sigma) := \mathrm{Pr}_{F \in \mu_p^M}\big[\text{exactly one of } F \cup \{\sigma\},\ F \setminus \{\sigma\} \text{ is in } \mathcal{F}\big].$$

As before, the superscript will often be omitted. The average sensitivity of a family is defined as sum of the influences of all elements.

$$\mathrm{as}_p(\mathcal{F}) := \sum_{\sigma \in M} \mathrm{Influence}_p(\mathcal{F}, \sigma).$$

Definition 9.3.1 *A family $\mathcal{F} \subseteq 2^M$ is called a core-family with a core $C \subseteq M$ if there exists a family $\mathcal{H} \subseteq 2^C$ such that*

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

A family $\mathcal{F} \subseteq 2^M$ is called *monotone* if $F \in \mathcal{F}$ and $F \subseteq F'$ implies $F' \in \mathcal{F}$. The following well-known lemma states that the weight of a monotone family is an increasing function of the bias parameter.

Lemma 9.3.2 *If $\mathcal{F} \subseteq 2^M$ is monotone and $p \geq q$, then $\mu_p^M(\mathcal{F}) \geq \mu_q^M(\mathcal{F})$.*

Russo-Margulis' Theorem in fact states that the weight of a monotone family is a continuous and differentiable function of the bias parameter and the derivative equals the average sensitivity of the family.

Theorem 9.3.3 *([94], [105]) For a monotone family $\mathcal{F} \subseteq 2^M$*

$$\left.\frac{d\mu_q(\mathcal{F})}{dq}\right|_{q=p} = \mathrm{as}_p(\mathcal{F})$$

Applying this Theorem and the Mean Value Theorem from analysis, we obtain

Lemma 9.3.4 *Let $\mathcal{F} \subseteq 2^M$ be a monotone family and $0 \leq p < p + \epsilon \leq 1$. Then there exists $p' \in (p, p+\epsilon)$ such that $\mathrm{as}_{p'}(\mathcal{F}) \leq \frac{1}{\epsilon}$.*

Proof: The Mean Value Theorem guarantees existence of $p' \in (p, p+\epsilon)$ such that

$$\mathrm{as}_{p'}(\mathcal{F}) = \left.\frac{d\mu_q(\mathcal{F})}{dq}\right|_{q=p'} = \frac{\mu_{p+\epsilon}(\mathcal{F}) - \mu_p(\mathcal{F})}{\epsilon} \leq \frac{1}{\epsilon}$$

■

Friedgut's Theorem [44] states that any family with low average sensitivity is essentially a core family. To be precise,

Theorem 9.3.5 *[44] Let $\mathcal{F} \subseteq 2^M$, $0 < p < 1$ be a bias parameter, η be an accuracy parameter and $k = \mathrm{as}_p(\mathcal{F})$. Then there exists a core family $\widehat{\mathcal{F}}$ with core $C \subseteq M$ such that*

- *Size of C is a constant that depends only on p, k, η. In fact $|C| \leq c_p^{k/\eta}$ where c_p depends only on p.*

- $\mu_p(\mathcal{F}\Delta\widehat{\mathcal{F}}) \leq \eta$ *where Δ denotes the symmetric difference of two families.*

Combining Lemma 9.3.4 and Theorem 9.3.5, we get :

Theorem 9.3.6 *Let p be a bias parameter, $\epsilon > 0$ be a constant and η be an "accuracy parameter". Let $\mathcal{F} \subseteq 2^M$ be a monotone family. Then there exists $p' \in (p, p+\epsilon)$ and a core family $\widehat{\mathcal{F}} \subseteq 2^M$ with a core $C \subseteq M$ such that*

- $\mathrm{as}_{p'}(\mathcal{F}) \leq \frac{1}{\epsilon}$.

- *The size of C is a constant that depends only on p, ϵ, η.*

- $\mu_{p'}(\mathcal{F}\Delta\widehat{\mathcal{F}}) < \eta$

Lemma 9.3.7 *Let $\mathcal{F} \subseteq 2^M$ be a monotone family. Let T be a set of elements such that for every element $\sigma \in T$, $\mathrm{Influence}_p(\mathcal{F}, \sigma) < \eta$. Define a subfamily $\mathcal{F}' \subseteq \mathcal{F}$ as*

$$\mathcal{F}' := \{F \mid F \in \mathcal{F},\ F \setminus T \in \mathcal{F}\}.$$

Then for any $0 < p < 1$ we have

$$\mu_p(\mathcal{F}') \geq \mu_p(\mathcal{F}) - \eta\, |T|\, (\min(p, 1-p))^{-|T|}.$$

Proof: A similar proof appears in [32]. Consider the family

$$\mathcal{F}'' := \{F \in 2^{M\setminus T} \mid F \cup T \in \mathcal{F},\ F \notin \mathcal{F}\}.$$

It can be seen that

$$\mu_p^M(\mathcal{F}) - \mu_p^M(\mathcal{F}') \leq \mu_p^{M\setminus T}(\mathcal{F}'').$$

For any set $F \in \mathcal{F}''$ there must exist some $D \subseteq T$ and an element $\sigma \in T$ such that $F \cup D \cup \{\sigma\} \in \mathcal{F}$ but $F \cup D \notin \mathcal{F}$. Hence, any set $F \in \mathcal{F}''$ contributes at least $\mu_p^{M\setminus T}(F) \cdot \min(p, 1-p)^{|T|}$ to the influence of one $\sigma \in T$. It remains to notice that the total influence of elements in T is at most $|T| \cdot \eta$. ■

The following lemma can be found as Lemma A.4 in [28].

Lemma 9.3.8 *Let $\epsilon > 0$ be an arbitrarily small constant and define $p = 1 - \frac{1}{k} - \epsilon$ to be the bias parameter. Then, for a sufficiently large universe M, the following holds. For any $\mathcal{F} \subseteq 2^M$ such that $\mu_p(\mathcal{F}) \geq 1 - \frac{1}{k}$ there exist k sets in the family $\mathcal{F}$ whose intersection is empty.*

9.4 Reduction to Vertex Cover in k-Uniform Hypergraphs

Let $\mathcal{L} = (G(V, W, E), \{\pi^{v,w}\})$ be an instance of Strong-LC given by Theorem 9.2.2 with parameters ζ, γ which will be chosen later. We will reduce this instance to an Independent Set problem on k-uniform hypergraphs. The vertices of the hypergraph we construct are weighted. One can obtain an unweighted hypergraph by using standard techniques (see [32]). The hypergraph will either contain an independent set of weight $1 - \frac{1}{k} - 2\epsilon$ or no independent set of weight δ where ϵ, δ can be made arbitrarily small. In the following we fix ϵ and δ and let $p = 1 - \frac{1}{k} - \epsilon$ be the bias parameter.

9.4.1 Construction of the Hypergraph

The set of vertices of the hypergraph will correspond to the bits of the long codes of labels assigned to vertices in V. Namely, the set of vertices is defined to be $V \times 2^M$. A vertex is a pair (v, F) where $v \in V$ is a vertex of the Strong-LC and $F \in 2^M$ is a subset of M. We define the *block* of vertices $B[v]$ for $v \in V$ as

$$B[v] := \{(v, F) \mid F \subseteq M\}$$

The weight of a vertex (v, F) is defined to be

$$\text{weight}(v, F) := \frac{1}{|V|} \cdot \mu_p^M(F)$$

Thus the sum of weights of all vertices in the hypergraph equals 1.

Now we define the edges of the hypergraph. For any two edges $(v_1, w), (v_2, w) \in E$ with common endpoint in W, and corresponding projections $\pi^{v_1,w}, \pi^{v_2,w}$, we define the following (hyper-)edges between the block $B[v_1]$ and the block $B[v_2]$:

$$\left\{ \{(v_1, I), (v_2, F_1), (v_2, F_2), \ldots, (v_2, F_{k-1})\} \;\middle|\; (\pi^{v_1,w})^{-1}(I) \cap (\pi^{v_2,w})^{-1}(\cap_{i=1}^{k-1} F_i) = \emptyset \right\}.$$

We say that these edges correspond to the pair of edges $(v_1, w), (v_2, w)$. Notice that every edge contains exactly k vertices, one vertex from the block $B[v_1]$ and $k - 1$ vertices from the block $B[v_2]$. Also note that we can have edges between $B[v_1]$ and $B[v_2]$ that correspond to more than one pair of edges. This can be as a result of parallel edges in $G(V, W, E)$ or as a result of several w-vertices to which both v_1 and v_2 are connected.

Completeness

Assume that the Strong-LC instance $\mathcal{L}$ has a labeling Φ in which at least $1-\zeta$ fraction of the V-vertices have all their edges satisfied. Let V_0 be the set of all such vertices with $|V_0| \geq (1-\zeta)|V|$. We claim that the following is an independent set:

$$\mathcal{IS} = \{(v, F) \mid v \in V_0,\ \Phi(v) \in F\}.$$

Consider any edge $\{(v_1, I), (v_2, F_1), \ldots, (v_2, F_{k-1})\}$ and let (v_1, w) and (v_2, w) be the pair of edges it corresponds to. Assume towards contradiction that all its vertices are in $\mathcal{IS}$. Clearly, this implies that $v_1, v_2 \in V_0$ and $\Phi(v_1) \in I$, $\Phi(v_2) \in \cap_{i=1}^{k-1} F_i$. Also, since all edges incident to both v_1 and v_2 are satisfied by Φ, we have

$$\pi^{v_1,w}(\Phi(w)) = \Phi(v_1), \quad \pi^{v_2,w}(\Phi(w)) = \Phi(v_2)$$

Therefore, $\Phi(w) \in (\pi^{v_1,w})^{-1}(I) \cap (\pi^{v_2,w})^{-1}(\cap_{i=1}^{k-1} F_i)$. In particular, this implies that $(\pi^{v_1,w})^{-1}(I) \cap (\pi^{v_2,w})^{-1}(\cap_{i=1}^{k-1} F_i) \neq \emptyset$ and we reach a contradiction by recalling the construction of the edges.

Note that with the bias parameter $p = 1 - \frac{1}{k} - \epsilon$, for every $v \in V_0$ the weight of the set $\mathcal{IS} \cap B[v]$ is exactly p times the total weight of vertices in $B[v]$. Hence

$$\text{weight}(\mathcal{IS}) = (1-\zeta) \cdot (1 - \frac{1}{k} - \epsilon) \geq 1 - \frac{1}{k} - 2\epsilon$$

since ζ can be chosen to be arbitrarily small.

9.4.2 Soundness

Now assume that there is no labeling to the Strong-LC instance $\mathcal{L}$ that satisfies even a γ fraction of the edges. We will show that the hypergraph contains no independent set of size δ. Assume towards contradiction that the hypergraph contains an independent set $\mathcal{IS}$ of size δ. For every $v \in V$, let

$$\mathcal{F}[v] = \{F \mid F \subseteq M,\ (v, F) \in \mathcal{IS}\}$$

Let V^* be the set of vertices v such that $\mu_p^M(\mathcal{F}[v]) \geq \delta/2$, i.e., a weight of at least $\delta/2$ of the total weight in the block $B[v]$ belongs to the independent set $\mathcal{IS}$. By an averaging argument, we have $|V^*| \geq \delta|V|/2$.

We will associate a "small" set of labels $L[v] \subseteq M$ for every $v \in V^*$ such that this labeling satisfies a weaker notion of consistency. More precisely we prove that

Lemma 9.4.1 *Given $\mathcal{IS}$ and V^* as above, there exists a constant $h = h(k, \epsilon, \delta)$ and non-empty sets of labels $L[v] \subseteq M$ for every $v \in V^*$ such that*

- $\forall\, v \in V^*,\ |L[v]| \leq h$
- *For every two edges $(v_1, w), (v_2, w)$, sharing the same endpoint w, we have*

$$(\pi^{v_1,w})^{-1}(L[v_1]) \cap (\pi^{v_2,w})^{-1}(L[v_2]) \neq \emptyset.$$

This is the main technical lemma in the analysis and we prove it in the next section. Let us see how this lemma is sufficient to arrive at a contradiction. The idea is to define one label for every vertex in $V \cup W$ such that this labeling satisfies more than a γ fraction of the edges.

We will try to satisfy only those edges which are incident to V^*. This is a $\delta/2$ fraction of all the edges since $|V^*| \geq \delta|V|/2$ and the bipartite graph $G(V, W, E)$ is left regular. Let W^* be the set of vertices in W which have an edge with some vertex in V^*. For every $w \in W^*$, fix $v(w)$ to be one vertex in V^* with which w has an edge. Define

$$L[w] := (\pi^{v(w),w})^{-1}(L[v(w)])$$

We claim that for every edge (v, w) with $v \in V^*$ and $w \in W^*$ we have,

$$\pi^{v,w}(L[w]) \cap L[v] \neq \emptyset.$$

When $v = v(w)$ this is clearly true and otherwise, it follows from Lemma 9.4.1.

Now consider the following probabilistic way of defining one label for every vertex in $V^* \cup W^*$. For $v \in V^*$ (resp. $w \in W^*$), define its label $\Phi(v)$ (resp. $\Phi(w)$) to be a randomly picked element of $L[v]$ (resp. $L[w]$). For each edge (v, w) with $v \in V^*$ and $w \in W^*$, the sets $\pi^{v,w}(L[w])$ and $L[v]$ intersect and both sets have size at most h. Therefore, with probability $1/h^2$, we have $\pi^{v,w}(\Phi(w)) = \Phi(v)$ and the edge is satisfied. Therefore, the expected fraction of satisfied edges is at least $\delta/(2h^2)$ and hence there must exist one labeling that satisfies these many edges. Choosing the parameter $\gamma < \delta/(2h^2)$ gives a contradiction.

9.4.3 Proof of Lemma 9.4.1

The set $L[v]$ for $v \in V^*$ will be constructed from the family $\mathcal{F}[v]$. Roughly speaking, the set $L[v]$ will be the core of the family $\mathcal{F}[v]$ along with all elements which have non-negligible influence on $\mathcal{F}[v]$.[1] Recall that for $v \in V^*$, we have $\mu_p^M(\mathcal{F}[v]) \geq \delta/2$.

Let $\eta > 0$ be a sufficiently small accuracy parameter which will be fixed later. Applying Theorem 9.3.6, we get

Lemma 9.4.2 *For every $v \in V^*$, there exists a real number $p[v] \in (1 - \frac{1}{k} - \epsilon, 1 - \frac{1}{k} - \frac{\epsilon}{2})$ and a core-family $\widehat{\mathcal{F}}[v] \subseteq 2^M$ with core $C[v]$ such that*

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

Let $\eta' > 0$ be a threshold parameter which will be chosen later. For every $v \in V^*$, we identify a set of elements Infl$[v] \subseteq M \setminus C[v]$ that have non-negligible influence on the family $\mathcal{F}[v]$, i.e.,

$$\text{Infl}[v] = \{\sigma \in M \setminus C[v] \mid \text{Influence}_{p[v]}(\mathcal{F}[v], \sigma) \geq \eta'\}.$$

Since $\mathcal{F}[v]$ has average sensitivity at most $\frac{2}{\epsilon}$ and the average sensitivity is simply the sum of influences of all the elements, it follows that the size of Infl$[v]$ is at most $\frac{2}{\eta'\epsilon}$ which is a constant. Finally, we define the set $L[v]$ as

$$L[v] := C[v] \cup \text{Infl}[v]. \tag{9.1}$$

[1]In [32], this set is referred to as the *extended core*.

Clearly, $L[v]$ has size at most $h := h_0 + \frac{2}{\eta' \epsilon}$.

To finish the proof of Lemma 9.4.1, it remains to show that for every pair of edges $(v_1, w), (v_2, w)$, we have $(\pi^{v_1,w})^{-1}(L[v_1]) \cap (\pi^{v_2,w})^{-1}(L[v_2]) \neq \emptyset$. Note that $\pi^{v_1,w}, \pi^{v_2,w}$ are bijections and w.l.o.g. we can assume them to be identity maps. Thus we need to show that $L[v_1] \cap L[v_2] \neq \emptyset$. It will be clear how the proof would work in the general case.

We will assume on the contrary that $L[v_1] \cap L[v_2] = \emptyset$ and reach a contradiction by exhibiting an edge $\{(v_1, I), (v_2, F_i)_{i=1}^{k-1}\}$ all of whose vertices are in the supposed independent set $\mathcal{IS}$. Let us begin with a lemma (admittedly the proof is cumbersome and could be skipped).

Lemma 9.4.3 *There exists $U_0 \subseteq C[v_1]$ such that defining $M' := M \setminus (C[v_1] \cup C[v_2])$ and $\mathcal{H}[v_1] \subseteq 2^{M'}$ as*

$$\mathcal{H}[v_1] := \{H \mid H \in 2^{M'},\ U_0 \cup H \in \mathcal{F}[v_1]\ \}$$

we have $\mu^{M'}_{p[v_1]}(\mathcal{H}[v_1]) \geq 1 - 8\eta/\delta$.

Proof: The assumption $L[v_1] \cap L[v_2] = \emptyset$ along with Equation (9.1) gives

$$C[v_1] \cap C[v_2] = \emptyset, \quad C[v_2] \cap \text{Infl}[v_1] = \emptyset.$$

This implies that every element of $C[v_2]$ has influence at most η' on the family $\mathcal{F}[v_1]$. Let $\mathcal{F}'[v_1] \subseteq \mathcal{F}[v_1]$ be a family defined as

$$\mathcal{F}'[v_1] := \{F \mid F \in \mathcal{F}[v_1], F \setminus C[v_2] \in \mathcal{F}[v_1]\}.$$

Applying Lemma 9.3.7, we get

$$\begin{aligned}\mu^M_{p[v_1]}(\mathcal{F}'[v_1]\,\Delta\,\mathcal{F}[v_1]) \quad &< \quad \eta'\,|C[v_2]|\,(\min(p[v_1],1-p[v_1]))^{-|C[v_2]|} \le \\ &\quad\ \ \eta'\,h_0\,(\min(p[v_1],1-p[v_1]))^{-h_0} \le \eta\end{aligned}$$

by choosing η' small enough. It follows that

$$\mu^M_{p[v_1]}(\widehat{\mathcal{F}}[v_1]\ \setminus\ \mathcal{F}'[v_1]) \le \mu^M_{p[v_1]}(\mathcal{F}'[v_1]\,\Delta\,\widehat{\mathcal{F}}[v_1])\ < 2\eta.$$

We would like to find a set $U_0 \subseteq C[v_1]$ in the core family $\widehat{\mathcal{F}}[v_1]$ such that the two families obtained by taking only the sets in $\widehat{\mathcal{F}}[v_1]$ and $\mathcal{F}'[v_1]$ whose intersection with $C[v_1]$ is U_0 are very close. We have,

$$\begin{aligned}2\eta \quad > \quad & \mu^M_{p[v_1]}(\widehat{\mathcal{F}}[v_1]\ \setminus\ \mathcal{F}'[v_1]) = \mathrm{Pr}_{D\in\mu^M_{p[v_1]}}[D \in \widehat{\mathcal{F}}[v_1]\ \setminus\ \mathcal{F}'[v_1]] = \\ & \sum_{U\subseteq C[v_1]} \mathrm{Pr}_{D\in\mu^M_{p[v_1]}}[D\cap C[v_1] = U \text{ and } D \in \widehat{\mathcal{F}}[v_1]\ \setminus\ \mathcal{F}'[v_1]] \overset{\{1\}}{=} \\ & \sum_{U\subseteq C[v_1],\ U\in\widehat{\mathcal{F}}[v_1]} \mu^{C[v_1]}_{p[v_1]}(U)\ \mathrm{Pr}_{D\in\mu^{M\setminus C[v_1]}_{p[v_1]}}[\ (U\cup D)\notin \mathcal{F}'[v_1]\ ]\end{aligned}$$

where $\{1\}$ holds since $\widehat{\mathcal{F}}[v_1]$ is a core family and hence depends only on $C[v_1]$. Since $\mu^{C[v_1]}_{p[v_1]}(\{U \subseteq C[v_1] \mid U \in \widehat{\mathcal{F}}[v_1]\}) \ge \delta/4$ this implies that there exists $U_0 \subseteq C[v_1]$, $U_0 \in \widehat{\mathcal{F}}[v_1]$ such that $\mathrm{Pr}_{D\in\mu^{M\setminus C[v_1]}_{p[v_1]}}[\ (U_0\cup D)\notin\mathcal{F}'[v_1]\ ] < 2\eta/(\delta/4)$. In other words, if we define $\mathcal{G}$ as $\{D \subseteq M\setminus C[v_1] \mid U_0\cup D \in \mathcal{F}'[v_1]\}$, then $\mu^{M\setminus C[v_1]}_{p[v_1]}(\mathcal{G}) > 1 - 8\eta/\delta$.

Finally, notice that $\mathcal{G}$ does not depend on $C[v_2]$. Hence, the family

$$\mathcal{H}[v_1] := \{H \mid H \subseteq M' = M\setminus(C[v_1]\cup C[v_2]),\ \ H\in\mathcal{G}\ \}$$

satisfies $\mu_{p[v_1]}^{M'}(\mathcal{H}[v_1]) = \mu_{p[v_1]}^{M \setminus C[v_1]}(\mathcal{G}) > 1 - 8\eta/\delta$, as required. ∎

Analogous to Lemma 9.4.3 we have by symmetry,

Lemma 9.4.4 *There exist* $V_0 \subseteq C[v_2]$ *such that defining* $M' := M \setminus (C[v_1] \cup C[v_2])$ *and* $\mathcal{H}[v_2] \subseteq 2^{M'}$ *as*

$$\mathcal{H}[v_2] := \{ H \mid H \in 2^{M'},\ V_0 \cup H \in \mathcal{F}[v_2] \ \}$$

we have $\mu_{p[v_2]}^{M'}(\mathcal{H}[v_2]) \geq 1 - 8\eta/\delta$.

Let $p^* := 1 - \frac{1}{k} - \frac{\epsilon}{2}$. Note that $\mathcal{H}[v_1]$ and $\mathcal{H}[v_2]$ are both monotone subfamilies of $2^{M'}$. Therefore, according to Lemma 9.3.2, $\mu_{p^*}^{M'}(\mathcal{H}[v_1]) \geq \mu_{p[v_1]}^{M'}(\mathcal{H}[v_1]) \geq 1 - 8\eta/\delta$ and similarly for v_2. Hence, the intersection of the families $\mathcal{H}[v_1]$ and $\mathcal{H}[v_2]$ satisfies

$$\mu_{p^*}^{M'}(\mathcal{H}[v_1] \cap \mathcal{H}[v_2]) \geq 1 - 16\eta/\delta > 1 - \frac{1}{k}$$

by picking η small enough. Hence, Lemma 9.3.8 implies that there exist sets $H_1, H_2, \ldots, H_k \in \mathcal{H}[v_1] \cap \mathcal{H}[v_2]$ such that

$$\cap_{i=1}^{k} H_i = \emptyset.$$

In particular, $H_1, H_2, \ldots, H_{k-1} \in \mathcal{H}[v_2]$ and $H_k \in \mathcal{H}[v_1]$.

Now define $I = U_0 \cup H_k$ and $F_i = V_0 \cup H_i$ for $1 \leq i \leq k-1$. By definition of the families $\mathcal{H}[v_1], \mathcal{H}[v_2]$, we have, $I \in \mathcal{F}[v_1], F_i \in \mathcal{F}[v_2]$ for $1 \leq i \leq k-1$. Thus $\{(v_1, I), (v_2, F_i)_{i=1}^{k-1}\}$ are vertices in the supposed independent set and they form an edge since

$$I \cap \left(\cap_{i=1}^{k-1} F_i \right) = \cap_{i=1}^{k} H_i = \emptyset.$$

This completes the proof.

Chapter 10

Hardness of Clique and Chromatic Number

Hardness results for clique are central to the theory of inapproximability. The seminal paper of Feige, Goldwasser, Lovász, Safra and Szegedy [38] obtained first hardness result for clique, and established the connection between PCPs and inapproximability. A long sequence of work (see [38], [13], [12], [17], [16], [59]) finally culminated in Håstad's result that clique is hard to approximate within factor $n^{1-\epsilon}$.

In this chapter, we further improve the hardness factor to $\frac{n}{2^{(\log n)^\gamma}}$ for some constant $\gamma < 1$. This takes us one step closer to the right answer for clique. The result is also interesting in light of Feige's result [36] on Lovász θ-function and Trevisan's result [111] on hardness of clique on bounded degree graphs.

We also obtain a similar hardness result for a related problem of finding the chromatic number of a graph. Main technique in this chapter is a Hadamard Code (as opposed to long code) based PCP which yields proofs of much smaller size. The result for chromatic number is shown via a technique called *Randomized PCP* introduced in [40]. It is much easier to apply this technique to PCPs in this chapter than earlier PCPs like Håstad's clique PCP [59].

10.1 Results and Techniques

The problem of finding the maximum size of a clique in an n-vertex graph is a well-studied NP-hard problem. The best known (polynomial time) approximation algorithm (see [21]) for MaxClique achieves an approximation ratio of $O(\frac{n}{\log^2 n})$ which suggests that this problem might be very hard to approximate. The first step towards proving strong inapproximability result for MaxClique was taken in a seminal paper by Feige et al. [38] who discovered the connection between PCPs and inapproximability of MaxClique. They were able to show a hardness factor of $2^{\log^{1-\epsilon} n}$ for an arbitrarily small constant $\epsilon > 0$. The discovery of PCP Theorem ([12], [13]) implied that MaxClique is inapproximable within factor n^c for some constant $c > 0$ unless NP = ZPP.

Bellare and Sudan [17] defined an important parameter of PCPs called *amortized free bit complexity* (recall Definition 3.1.3) and showed that

Theorem 10.1.1 *If* NP *has a PCP verifier that uses logarithmic randomness, has completeness* $\geq \frac{1}{2}$ *and amortized free bit complexity* $\overline{f}$*, then assuming* $\mathrm{NP} \not\subseteq \mathrm{BPP}$*, no polynomial time algorithm can approximate clique size in an* n*-vertex graph within factor* $n^{\frac{1}{1+\overline{f}}-\epsilon}$*. Here* $\epsilon > 0$ *is an arbitrarily small constant.*

They constructed PCPs with 3 amortized free bits and obtained a hardness factor of $n^{1/4-\epsilon}$ for clique. This was improved to $n^{1/3-\epsilon}$ by Bellare et al [16].

In his breakthrough result, Håstad [59] proved an $n^{1-\epsilon}$ inapproximability factor for MaxClique. He obtained a PCP verifier that achieves amortized free bit complexity δ for arbitrarily small constant $\delta > 0$. A simpler and alternate proof of this result was obtained by Samorodnitsky and Trevisan [106] which was further simplified recently by Håstad and Wigderson [64]. These verifiers are based on linearity testing algorithms and make a clever use of *recycling of queries* whose study was initiated by Trevisan [110]. They simultaneously achieve δ amortized free bits and $1 + \delta$ *amortized query bits* which is optimal. In Chapter 3 of this thesis, we saw such a verifier which in addition achieves perfect completeness.

Results

Feige [36] showed that one natural idea for approximating maximum clique-size, the polynomial time computable Lovász's θ-function, has approximation ratio as bad as $\frac{n}{2^{O(\sqrt{\log n})}}$. Arguably, Lovász's θ-function might provide the "best" approximation guarantee, so it is conceivable that $\frac{n}{2^{O(\sqrt{\log n})}}$ hardness factor could be shown for MaxClique. This intuition is supported by Trevisan's [111] $\frac{k}{2^{O(\sqrt{\log k})}}$ hardness for clique in degree-k graphs (k thought of as a constant). Can this result be interpolated all the way upto $\frac{n}{2^{O(\sqrt{\log n})}}$? As a step towards resolving this question, we show that

Theorem 10.1.2 *It is hard to approximate MaxClique in polynomial time within factor* $\frac{n}{2^{(\log n)^{1-\gamma}}}$ *for some constant* $\gamma > 0$ *unless* $NP \subseteq ZPTIME(2^{(\log n)^{O(1)}})$.

Chromatic number of a graph is defined to be the minimum number of colors needed to color the graph. It is NP-hard to find the chromatic number exactly (even to test if a graph is 3-colorable). Feige and Kilian [40] introduced a technique called *Randomized PCPs* and applied this technique to Håstad's clique PCP [59] to show that :

Theorem 10.1.3 *It is hard to approximate chromatic number of a graph in polynomial time within factor* $n^{1-\epsilon}$ *for any constant* $\epsilon > 0$ *unless* NP = ZPP.

Note that approximating chromatic number is different from Graph Coloring problem considered in Chapter 3. In the latter problem, we are given a graph with a *constant* chromatic number and we desire a coloring using few colors.

Randomizing Håstad's PCP is rather tedious. On the other hand, the PCP we construct in this chapter is very easy to randomize and also gives a stronger hardness factor. We prove that :

Theorem 10.1.4 *It is hard to approximate chromatic number of a graph in polynomial time within factor* $\frac{n}{2^{(\log n)^{1-\gamma}}}$ *for some constant* $\gamma > 0$ *unless NP* $\subseteq$ *ZPTIME*$(2^{(\log n)^{O(1)}})$.

Engebretsen and Holmerin [33] obtained $\frac{n}{2^{O(\log n/\sqrt{\log \log n})}}$ hardness factor for both clique and chromatic number, using Samorodnitsky and Trevisan's PCP [106]. Our improvement comes from a new PCP construction based on Hadamard codes.

Techniques

Most PCPs are based on the following standard paradigm. First construct the *Raz Verifier* (see Section 2.2) by parallel repetition of a basic 2-prover 1-round protocol for Gap-3SAT and then expect the provers' answers in some encoded form. In these constructions, one of the provers is supposed to return an assignment to a set of clauses and it is necessary to restrict his answer to a *satisfying* assignment to the clauses. This is achieved using one of the two methods : First, define long code over the domain of (only) satisfying assignments. Second, define long code over the domain of all assignments and use a technique called *conditioning* or *folding* [60]. Both methods require the use of Long Code due to non-linearity of 3SAT predicate. However the long code is extremely redundant. It encodes a u-bit string by a 2^{2^u}-bit string and requires too much randomness to check it probabilistically. This turns out to be a barrier in improving hardness result for MaxClique.

In this chapter, we present a PCP verifier based on Hadamard code which encodes a u-bit string by a 2^u-bit string and allows a more randomness efficient checking. We start with a Raz Verifier obtained by parallel repetition of a basic 2-prover 1-round protocol for Max-3-Lin-2 (see Theorem 2.0.2). This problem features a linear predicate on 3 bits. It turns out that the Hadamard code is powerful enough to fold over *linear* constraints and we can use Hadamard code instead of the long code. Apart from using a different code, our verifier is constructed and analyzed along the same lines as [106]. However our construction yields proofs of much smaller size.

Hardness of chromatic number is proved using a technique called *Randomized PCP* introduced by Feige and Kilian [40]. Usually, the completeness condition of PCP requires existence of one correct proof. This proof corresponds to a large independent set in the FGLSS graph. We now make a stronger demand : we require a collection of proofs such that the corresponding independent sets cover every vertex in FGLSS graph almost uniformly. Randomized PCPs allow us to achieve this property. This gives a FGLSS graph with low *fractional chromatic number*. Advantage of working with fractional chromatic number is that this parameter is multiplicative under inclusive graph product. The reduction consists of taking a multiple self-product of the FGLSS graph and then taking a random induced subgraph of appropriate size. The reduction appears in [40] and we give only a brief sketch in this chapter.

Overview of the Chapter :

Section 10.2 gives necessary background. Section 10.3 gives our main PCP construction. The hardness result for MaxClique follows directly from our PCP construction and it is proved in Section 10.4. Section 10.5 introduces randomized PCPs and Section 10.6 proves hardness result for chromatic number.

10.2 Preliminaries

This section explains some of the technical tools used in this chapter. Our goal is to construct PCPs based on Hadamard codes as opposed to long codes used in earlier chapters. Fourier analysis of Hadamard codes and corresponding notation is somewhat different. Hadamard codes are defined using linear functions, so we need to use an underlying NP-hard problem that features linear constraints, namely Max-3-Lin-2.

We restate Theorem 2.0.2 in a form convenient to us. We call an instance Γ of Max-3-Lin-2 regular if every equation contains exactly 3 variables and every variable appears in exactly the same number (say 7) of equations. We call the instance Γ given by the following theorem as an instance of Max-3-Lin-2(ϵ).

Theorem 10.2.1 *There exists an absolute constant $\mu < 1$ such that, for arbitrarily small constant $\epsilon > 0$, there exists a polynomial time reduction from a 3SAT formula ϕ with n variables to a regular Max-3-Lin-2 instance Γ with N variables such that : If ϕ is satisfiable, there exists an assignment to variables in Γ that satisfies $1-\epsilon$ fraction of equations. If ϕ is unsatisfiable, no assignment can satisfy more than μ fraction of the equations. Moreover, the reduction can achieve $\epsilon = \frac{1}{(\log N)^{\beta}}$ for some constant $\beta > 0$ if we allow the running time of the reduction and N to be slightly superpolynomial, i.e. $n^{O(\log\log n)}$.*

Remark : Theorem 2.0.2 actually gives a gap of $1-\epsilon$ versus $\frac{1}{2}+\delta$ where ϵ, δ are arbitrarily small. However the instance given by this reduction is not regular. It can be made regular in a similar manner as in the proof of Theorem 10.2 in [11, Chapter 10]. The soundness suffers in this regularization process, but is still bounded away from 1.

10.2.1 The Raz Verifier

As in earlier chapters, our construction makes use of the *Raz Verifier* which we define next. However we use a Raz Verifier based on Max-3-Lin-2(ϵ) rather than the standard one based on Gap-3SAT.

The Raz Verifier is obtained by parallel repetition of a basic 2-prover-1-round protocol and then expecting the provers' answers as a written proof. We directly describe the final construction. Let Γ be an instance of Max-3-Lin-2(ϵ) given by Theorem 10.2.1 and u be an integer parameter. The verifier expects to have two proofs P and Q, where proof P is supposed to contain for every set v of u variables, a u-bit string $P(v)$ giving the values of these variables in some (global) assignment. The proof Q is supposed to contain for every set w of u equations, a $3u$-bit string $Q(w)$ giving the values of the $3u$ variables occurring in these u equations. We will also denote by w the set of these $3u$ variables.

The Raz Verifier works as follows : It randomly picks variables $v = (x_i)_{i=1}^u$ and then picks equations $w = (C_i)_{i=1}^u$ where equation C_i is chosen randomly from the constantly many equations containing variable x_i. It reads the bit-strings $P(v)$ and $Q(w)$ respectively. Let π be the projection from $3u$-bit strings to u-bit strings which corresponds to restricting an assignment to the set w to an assignment to the set v. The verifier accepts if and only if both these tests are satisfied :

- (Linear constraints test :) $Q(w)$ satisfies all equations $(C_i)_{i=1}^u$.
- (Projection/Consistency test :) $P(v) = \pi(Q(w))$, i.e. values of variables $(x_i)_{i=1}^u$ in $P(v)$ and $Q(w)$ are the same.

Completeness of the Raz Verifier is $\geq (1-\epsilon)^u \geq 1-\epsilon u$. This is because if there is an assignment that satisfies $1-\epsilon$ fraction of the equations, both the proofs P and Q can be consistent with this assignment. With probability $(1-\epsilon)^u$, all the equations $(C_i)_{i=1}^u$ are satisfied and the verifier accepts. When at most $\mu < 1$ fraction of the equations in Γ are satisfiable, the soundness can be upper bounded by Raz's Parallel Repetition Theorem.

Theorem 10.2.2 *There exists an absolute constant $C_{lin} < 1$ such that the soundness of the Raz Verifier for Max-3-Lin-2(ϵ) is at most C_{lin}^u.*

10.2.2 Fourier Analysis

We use Fourier analysis for Hadamard codes as opposed to the Fourier analysis for long codes. Note the difference in notation : for example, α now denotes a vector whereas earlier α denoted a subset of the domain over which long code was defined.

Consider the vector space of all functions $A : \mathbb{F}_2^u \rightarrow \mathbb{R}$ where addition of two functions is defined as pointwise addition. The dimension of this space is 2^u. Define an inner product on this vector space as follows.

$$< A_1, A_2 >= \frac{1}{2^u} \sum_{a \in \mathbb{F}_2^u} A_1(a) \cdot A_2(a)$$

We will identify an orthonormal basis w.r.t. this inner product. A function $A : \mathbb{F}_2^u \rightarrow \{1, -1\}$ is called linear if $A(x \oplus y) = A(x) \cdot A(y)$, where $\oplus$ denotes vector addition over $\mathbb{F}_2$. There are precisely 2^u linear functions. For every $\alpha \in \mathbb{F}_2^u$, there is a function χ_α defined by

$$\chi_\alpha(a) = (-1)^{a \cdot \alpha} \quad \forall\, a \in \mathbb{F}_2^u$$

We have following easy lemmas.

Lemma 10.2.3 *For* $\alpha, \alpha' \in \mathbb{F}_2^{\mathrm{u}} \qquad \chi_\alpha(\mathrm{a}) \cdot \chi_{\alpha'}(\mathrm{a}) = \chi_{\alpha \oplus \alpha'}(\mathrm{a})$

Lemma 10.2.4 *For* $\alpha \in \mathbb{F}_2^{\mathrm{u}}$

$$E_a[\chi_\alpha(a)] = \begin{cases} 1 & \textit{if} \quad \alpha = 0 \\ 0 & \textit{if} \quad \alpha \neq 0 \end{cases}$$

It follows from these lemmas that the set of all linear functions is an orthonormal basis. Thus any function $A : \mathbb{F}_2^{\mathrm{u}} \to \{-1, 1\}$ can be uniquely expressed as $A = \sum_\alpha \widehat{A}_\alpha \chi_\alpha$ where $\widehat{A}_\alpha$ are its *Fourier coefficients* given by

$$\widehat{A}_\alpha = \frac{1}{2^u} \sum_{a \in \mathbb{F}_2^{\mathrm{u}}} A(a) \cdot \chi_\alpha(a)$$

The Fourier coefficients satisfy Parseval's identity $\sum_\alpha \widehat{A}_\alpha^2 = 1$.

A *projection function* $\pi : \mathbb{F}_2^{3\mathrm{u}} \to \mathbb{F}_2^{\mathrm{u}}$ is a function that maps vectors in $\mathbb{F}_2^{3\mathrm{u}}$ to some fixed u coordinates. For $a \in \mathbb{F}_2^{\mathrm{u}}$, let $\pi^{-1}(a)$ denote the unique vector $c \in \mathbb{F}_2^{3\mathrm{u}}$ such that $\pi(c) = a$ and the coordinates of c other than those projected by π are 0.

10.2.3 Hadamard Codes, their Decoding and Folding

Using the standard paradigm, our PCP verifier will expect an encoding of the proof supplied to the Raz Verifier. Specifically, we use Hadamard codes which we define next.

Definition 10.2.5 *Hadamard code of* $p \in \mathbb{F}_2^{\mathrm{u}}$ *is the* 2^u*-bit string* $\{\chi_p(a)\}_{a \in \mathbb{F}_2^{\mathrm{u}}}$. *We denote it by* $Hadamard(p)$.

Recall that the string $y = Q(w)$ read by the Raz Verifier is supposed to satisfy a set of u linear constraints. Let these constraints be : $h_1 \cdot y = \zeta_1, \ldots, h_u \cdot y = \zeta_u$ where $h_1, \ldots, h_u \in \mathbb{F}_2^{3\mathrm{u}}$ and $\zeta_1, \ldots, \zeta_u \in \mathbb{F}_2$. We called this the linear constraints test.

Folding : We use a technique called *folding* that enables the verifier to ignore the linear constraints test.

Suppose that B is Hadamard code of y and y satisfies the constraints mentioned above. Let H be the linear subspace spanned by the vectors $h_1, \dots, h_u$. Then for any vector b and any vector $h \in H,\ h = \oplus \rho_i h_i$, we have

$$B(b \oplus h) = (-1)^{y \cdot (b \oplus h)} = (-1)^{y \cdot b} \cdot (-1)^{y \cdot \sum_i \rho_i h_i} = B(b) \cdot (-1)^{\sum_i \rho_i y \cdot h_i} = B(b) \cdot (-1)^{\sum_i \rho_i \zeta_i}$$

Motivated by this observation, for an arbitrary function $B : \mathbb{F}_2^{3u} \to \{1, -1\}$, we define another function B' as :

$$\text{For } b = v_b \oplus_i \rho_i h_i,\ \ \rho_1, \dots, \rho_u \in \mathbb{F}_2, \quad \mathrm{B'(b) = B(v_b) \cdot (-1)^{\sum_i \rho_i \zeta_i}}$$

where v_b denotes the lexicographically smallest vector in the set of vectors $b \oplus H$ (the group theoretic coset of H). We call B' a folding of B over the linear constraints. A crucial consequence of folding is :

Lemma 10.2.6 *If $\widehat{B'}_\beta \neq 0$, then β must satisfy the linear constraints, i.e. $h_i \cdot \beta = \zeta_i \ \forall\, i$.*

Proof: Consider any particular constraint $h_i \cdot y = \zeta_i$. By definition,

$$\widehat{B'}_\beta = \frac{1}{2^{3u}} \sum_b B'(b) \cdot \chi_\beta(b) = \frac{1}{2^{3u+1}} \sum_b (B'(b) \cdot \chi_\beta(b) + B'(b \oplus h_i) \cdot \chi_\beta(b \oplus h_i))$$

$$= \frac{1}{2^{3u+1}} \sum_b B'(b) \cdot \chi_\beta(b)(1 + (-1)^{\zeta_i} \chi_\beta(h_i))$$

This sum is zero unless $h_i \cdot \beta = \zeta_i$. Thus β satisfies all the constraints if $\widehat{B'}_\beta \neq 0$. ∎

We can use appropriate access mechanism to force a function B to be folded. When the verifier wants to read a bit $B(b)$, it reads $B(v_b)$ instead and calculates the value of $B(b)$ from it. Thus we can assume that the B-tables in the proof are folded over respective linear constraints.

We will eventually show that if our PCP verifier accepts the encoded proofs with a good probability, then these proofs can be decoded to construct proofs (P, Q) which the Raz Verifier accepts with a good probability. Decoding of a table B gives β with probability $\widehat{B}_\beta^2$. (Since $\sum_\beta \widehat{B}_\beta^2 = 1$, this defines a valid probability distribution). Folding ensures that any β given by this decoding procedure satisfies the linear constraints on $Q(w)$. Thus folding enables us to forget about the linear constraints test and focus only on the projection test while analyzing our PCP construction.

We point out again that the previous PCP constructions use Gap-3SAT as the underlying NP-hard problem where the constraints are non-linear and one cannot use Hadamard codes.

10.3 The Main PCP Construction and Analysis

We now define and analyze our PCP verifier which we call V_{lin}. Apart from the use of Hadamard code, it is similar to Samorodnitsky and Trevisan's verifier [106] and analyzed in a similar manner.

The verifier V_{lin} is given an instance Γ of Max-3-Lin-2(ϵ). As mentioned before, it expects to have proofs (P', Q') which are Hadamard encodings of proofs (P, Q) for the Raz Verifier. So $P'(v)$ ($Q'(w)$) is now supposed to contain the Hadamard code of string $P(v)$ ($Q(w)$). The verifier V_{lin} proceeds as follows :

1. Pick a set v of u variables at random and k sets $(w_j)_{j=1}^k$ independently, where each set w_j is picked in a similar manner as the Raz Verifier. Let π_j be the projection function between w_j and v.

2. Let A be the supposed Hadamard code of $P(v)$ and B_j be the supposed Hadamard code of $Q(w_j)$ in the proof. Tables B_j are assumed to be folded over respective linear constraints.

3. Pick $a_1, \ldots, a_k \in \mathbb{F}_2^{\mathrm{u}}$ and $b_1, \ldots, b_k \in \mathbb{F}_2^{3\mathrm{u}}$ randomly.

4. Accept if and only if for $1 \leq i, j \leq k$

$$A(a_i)B_j(b_j) = B_j(\pi_j^{-1}(a_i) \oplus b_j) \tag{10.1}$$

Recall that for a projection $\pi : \mathbb{F}_2^{3\mathrm{u}} \to \mathbb{F}_2^{\mathrm{u}}$ and $a \in \mathbb{F}_2^{\mathrm{u}}$, $\pi^{-1}(a)$ is the unique vector $c \in \mathbb{F}_2^{3\mathrm{u}}$ such that $\pi(c) = a$ and the coordinates of c other than those projected by π are 0. It can be easily checked that for any linear function $\chi_\beta,\ \beta \in \mathbb{F}_2^{3\mathrm{u}}$, we have

$$\chi_\beta(\pi^{-1}(a)) = \chi_{\pi(\beta)}(a)$$

We will prove the following theorem giving the properties of the verifier V_{lin}.

Theorem 10.3.1 *The verifier V_{lin} for Max-3-Lin-2(ϵ) instance Γ with N variables*

- *Uses $r = u \log N + O(ku)$ random bits.*

- *Queries $2k + k^2$ bits from the proof with $f = 2k$ free bits.*

- *Has completeness at least $1 - \epsilon ku$.*

- *Has soundness* $2^{-k^2} + \delta$ *provided* $C^u_{lin} < \delta^2$.

Proof: The claims about the number of random bits and the number of query bits are clear. After reading the $2k$ bits $\{A(a_i), B_j(b_j)\}_{i,j=1}^k$, the answers of the remaining k^2 queries are uniquely determined by Equation (10.1). Hence the number of free bits is $2k$. For the completeness, we know that the instance Γ has an assignment satisfying $1-\epsilon$ fraction of the equations. Consider a proof which is consistent with this assignment and is encoded using correct Hadamard codes. The verifier picks ku equations in total and each equation is picked uniformly at random. So with probability $1-\epsilon ku$, all the equations are satisfied. In the constructed proof, A is a Hadamard code of some $x \in \mathbb{F}_2^{\mathrm{u}}$ and B_j is a Hadamard code of some $y_j \in \mathbb{F}_2^{3\mathrm{u}}$ with $\pi_j(y_j) = x$. Thus for all $1 \le i, j \le k$

$$\begin{aligned} B_j(\pi_j^{-1}(a_i) \oplus b_j) &= (-1)^{y_j \cdot \pi_j^{-1}(a_i) \,\oplus\, y_j \cdot b_j} \\ &= (-1)^{\pi_j(y_j)\cdot a_i} \cdot (-1)^{y_j \cdot b_j} \\ &= (-1)^{x \cdot a_i} \cdot B_j(b_j) \\ &= A(a_i) B_j(b_j) \end{aligned}$$

Thus the test accepts with probability $1-\epsilon ku$. The main task is to prove soundness. We will show that if the soundness is $2^{-k^2} + \delta$ then there exist proofs (P, Q) which the Raz Verifier accepts with probability δ^2. This will contradict the fact that the Raz Verifier has soundness at most C^u_{lin} (see Theorem 10.2.2).

Let $Acc(i,j) = A(a_i)B_j(b_j)B_j(\pi_j^{-1}(a_i) \oplus b_j)$ and consider the following expression

$$\prod_{i,j=1}^{k} \frac{1 + Acc(i,j)}{2}$$

Clearly, this expression is 1 if the test accepts and 0 otherwise. So the acceptance probability of the verifier is expectation of this expression over the choice of $v, \{a_i\}_{i=1}^k, \{w_j, b_j\}_{j=1}^k$. Expanding out the product, the acceptance probability is given by

$$\frac{1}{2^{k^2}} \sum_{S \subseteq [k]\times[k]} T_S \quad \text{where} \quad T_S = E_{\substack{v,w_1,\dots,w_k;\\ a_1,\dots,a_k,b_1,\dots,b_k}} \left[\prod_{(i,j)\in S} Acc(i,j) \right]$$

If this probability is $\geq 2^{-k^2} + \delta$, there exists a nonempty set $S \subseteq [k] \times [k]$ such that $T_S \geq \delta$. The following ingenious lemma by Samorodnitsky and Trevisan [106] enables us to assume that S is of the form $[2] \times [d]$ for some $1 \leq d \leq k$. We provide a proof of this lemma to make our presentation self-contained.

Lemma 10.3.2 *If $T_S \geq \delta$ for some non-empty set $S \subseteq [k] \times [k]$, then $T_{[2]\times[d]} \geq \delta^2$ for some $1 \leq d \leq k$.*

Proof: Suppose without loss of generality that $(1,1) \in S$ and $(1,2), \dots (1,d)$ are other pairs in S with the first coordinate 1. We fix $v, w_1, \dots, w_k$ for the time being. Let us divide our random choice of $(a_i, b_j)_{i,j=1}^k$ into X given by the choice of (a_1, b_1), and Y given by the choice of the rest. Let S_1 be the subset of S containing $(1,1), (1,2) \dots (1,d)$. Then

$$E_{X,Y}[\prod_{(i,j)\in S} \mathrm{Acc}(i,j)] = E_{X,Y}[\prod_{j=1}^{d} \mathrm{Acc}(1,j) \cdot \prod_{(i,j)\in S\setminus S_1} \mathrm{Acc}(i,j)] =$$
$$E_{X,Y}[F(X,Y)G(Y)] = E_Y[E_X[F(X,Y)]G(Y)]$$

for some functions F and G with values in $\{-1,1\}$. Applying Cauchy-Schwartz inequality this can be bounded by

$$\sqrt{E_Y[(E_X[F(X,Y)])^2]}\sqrt{E_Y[G(Y)^2]} \leq \sqrt{E_Y[(E_X[F(X,Y)])^2]} =$$

$$\sqrt{E_Y[E_{X_1}[F(X_1,Y)] \cdot E_{X_2}[F(X_2,Y)]]} = \sqrt{E_{X_1,X_2,Y}[F(X_1,Y) \cdot F(X_2,Y)]}$$

where X_1, X_2 are identically distributed as X and are independent. However the term $F(X_1,Y) \cdot F(X_2,Y)$ is equal to

$$\prod_{j=1}^{d} Acc(1,j) \cdot \prod_{j=1}^{d} Acc(2,j) = \prod_{(i,j) \in S' = [2]\times[d]} Acc(i,j)$$

Considering expectation over $v, w_1, \dots, w_k$ we get

$$\begin{aligned}
\delta \ \le \ T_S \ &= \ E_{v,w_1,\dots,w_k}\left[E_{X,Y}[\prod_{(i,j)\in S} Acc(i,j)\ ] \right] \\
&\le \ E_{v,w_1,\dots,w_k}\left[\sqrt{E_{X,Y}[\prod_{(i,j)\in S'} Acc(i,j)]} \right] \\
&\le \ \sqrt{E_{v,w_1,\dots,w_k,X,Y}[\prod_{(i,j)\in S'} Acc(i,j)]} = \sqrt{T_{S'}}
\end{aligned}$$

■

We first consider the case when $S = [2] \times [d]$ and d is even. In this case

$$T_S = E\left[\prod_{i=1}^{2} \prod_{j=1}^{d} A(a_i) B_j(b_j) B_j(\pi_j^{-1}(a_i) \oplus b_j) \right]$$

In this product $A(a_1), A(a_2)$ and $B_j(b_j)$ appear an even number of times. Since these values are ± 1, the product reduces to

$$T_S = E\left[\prod_{j\in[d]} B_j(\pi_j^{-1}(a_1) \oplus b_j) B_j(\pi_j^{-1}(a_2) \oplus b_j) \right]$$

We substitute the following Fourier expansions

$$\begin{aligned}
B_j(\pi_j^{-1}(a_1) \oplus b_j) &= \sum_{\beta_j} \widehat{B}_{j\beta_j} \chi_{\beta_j}(\pi_j^{-1}(a_1) \oplus b_j) \\
&= \sum_{\beta_j} \widehat{B}_{j\beta_j} \chi_{\pi_j(\beta_j)}(a_1) \chi_{\beta_j}(b_j) \\
B_j(\pi_j^{-1}(a_2) \oplus b_j) &= \sum_{\gamma_j} \widehat{B}_{j\gamma_j} \chi_{\pi_j(\gamma_j)}(a_2) \chi_{\gamma_j}(b_j)
\end{aligned}$$

where we used the fact that $\chi_\beta(\pi^{-1}(a)) = \chi_{\pi(\beta)}(a)$. Expanding and using Lemma 10.2.3 we get

$$T_S = \sum_{\beta_j, \gamma_j, j \in [d]} \prod_{j \in [d]} \widehat{B}_{j\beta_j} \widehat{B}_{j\gamma_j} \;\; E\left[\chi_{\oplus_j \pi_j(\beta_j)}(a_1) \cdot \chi_{\oplus_j \pi_j(\gamma_j)}(a_2) \cdot \prod_{j \in [d]} \chi_{\beta_j \oplus \gamma_j}(b_j) \right]$$

Taking expectation over b_j, from Lemma 10.2.4, we see that the terms in this summation are non-zero only if $\beta_j = \gamma_j \; \forall \; j$. Taking expectation over a_1, we see that we have nonzero terms only if $\oplus_{j \in [d]} \pi_j(\beta_j) = 0$. We conclude that

$$\delta^2 \;\leq\; \quad T_S = E_{v, w_1, \ldots, w_d}\left[\sum_{\beta_j :\; \oplus_{j \in [d]} \pi_j(\beta_j) = 0} \prod_{j=1}^{d} \widehat{B}^2_{j\beta_j} \right] \tag{10.2}$$

The case when $S = [2] \times [d]$ and d is odd is similar. In this case we have

$$T_S = E\left[A(a_1) A(a_2) \prod_{j \in [d]} B_j(\pi_j^{-1}(a_1) \oplus b_j) B_j(\pi_j^{-1}(a_2) \oplus b_j) \right]$$

Using Fourier expansions of $A, B_1, \ldots, B_d$ and simplifying, we get

$$\delta^2 \;\le\; T_S \;=\; E_{v,w_1,\ldots,w_d}\left[\sum_{\alpha,\ \beta_j:\ \alpha=\oplus_{j\in[d]}\pi_j(\beta_j)} \widehat{A}_\alpha^2 \prod_{j=1}^{d} \widehat{B}_{j\beta_j}^2\right] \qquad (10.3)$$

Now we define proofs (P, Q) for the Raz Verifier as follows. For a set w, pick β with probability $\widehat{B}_\beta^2$ and define $Q(w) = \beta$. For a set v, pick sets $(w_j)_{j=2}^d$ at random and pick $(\beta_j)_{j=2}^d$ with probability $\prod_{j=2}^d \widehat{B}_{j\beta_j}^2$. If d is even, define $P(v) = \oplus_{j=2}^d \pi_j(\beta_j)$. If d is odd, pick α with probability $\widehat{A}_\alpha^2$ and define $P(v) = \alpha \oplus_{j=2}^d \pi_j(\beta_j)$.

We claim that the acceptance probability of the Raz Verifier on these proofs (expected over construction of (P, Q)) is precisely the expressions in (10.2) or (10.3). Consider expression (10.2). $Q(w_1)$ will be defined to be β_1 with probability $\widehat{B}_{1\beta_1}^2$. After picking $(w_2, \ldots, w_d)$, $P(v)$ will be defined to be $\oplus_{j=2}^d \pi_j(\beta_j)$ with probability $\prod_{j=2}^d \widehat{B}_{j\beta_j}^2$. The condition $\oplus_{j\in[d]}\pi_j(\beta_j) = 0$ is equivalent to the condition $\pi_1(\beta_1) = \oplus_{j=2}^d \pi_j(\beta_j)$, i.e. $\pi_1(Q(w_1)) = P(v)$. Thus expression (10.2) can be rewritten as

$$\delta^2 \;\le\; E_{v,w_1}\left[\,\Pr[\,\pi(Q(w_1)) = P(v)\,]\,\right]$$

where the probability is taken over the construction of the proofs (P, Q). The condition $\pi(Q(w_1)) = P(v)$ is precisely the condition when the Raz Verifier accepts and the claim follows. The expression (10.3) corresponding to the case when d is odd can be handled similarly.

Thus there exists at least one choice of proofs (P, Q) which is accepted by the Raz Verifier with probability at least δ^2 concluding the proof of Theorem 10.3.1. ∎

As an immediate consequence of Theorem 10.3.1 we get

Theorem 10.3.3 *For any constants $\epsilon, \delta > 0$, NP has probabilistically checkable proof systems where the verifier uses logarithmic randomness, has completeness $1 - \epsilon$, amortized query complexity $1 + \delta$ and amortized free bit complexity δ.*

This theorem was first proved by Samorodnitsky and Treivsan [106]. This result is optimal since PCPs with amortized query complexity less than 1 can recognize languages only in BPP [16, Lemma 10.6].

10.4 Improved Inapproximability Result for MaxClique

In this section, we prove Theorem 10.1.2. We use the verifier V_{lin} from Theorem 10.3.1 with superconstant values of parameters u, k and subconstant value of ϵ as given by Theorem 10.2.1.

We construct a PCP verifier for 3SAT as follows : Using Theorem 10.2.1, we transform given 3SAT formula to an instance Γ of Max-3-Lin-2(ϵ) with $\epsilon = \frac{1}{(\log N)^{\beta}}$ and $N = n^{O(\log \log n)}$. Then we use Theorem 10.3.1 to construct a PCP verifier for Γ with parameters :

- $\epsilon = \frac{1}{(\log N)^{\beta}},\ u = \frac{1}{2}(\log N)^{3\beta/4},\ k = (\log N)^{\beta/4},\ f = 2k$
- $r \leq (\log N)^{1+3\beta/4}$
- $c \geq 1 - \epsilon k u \geq 1/2$
- $\delta = 2^{-k^2},\ \ s \leq 2 \cdot 2^{-k^2}$. The choice of $u,\ k$ ensures that $C^{u}_{lin} < \delta$.

Finally we use a well-known connection between PCPs and hardness of approximating clique size, discovered by Feige et al [38]. Their reduction reduces a PCP to a graph which is well-known as the FGLSS graph. A brief description of this reduction appears in

Section 3.4.1. Zuckerman [114] augmented the FGLSS reduction by a gap-amplification technique based on dispersers. The following theorem is implicit in [114] and [16] and can be found explicitly in [33, Lemma 6.3].

Theorem 10.4.1 *If there is a PCP verifier for 3SAT using r random bits, f free query bits, completeness c and soundness s, then for any $R > r$ and $D = (R+2)/\log(1/s)$, there is a randomized reduction from a 3SAT formula ϕ to a graph G with $N' = 2^{R+Df}$ vertices such that : If ϕ is satisfiable, with probability $2/3$, G has a clique of size at least $c^D 2^R/2$ and if ϕ is unsatisfiable, with probability $2/3$, maximum clique size in G is at most 2^r. This reduction runs in time polynomial in N' and the running time of the verifier.*

We take $R = r(\log N)^{\beta/4}$. Note that $N = n^{O(\log\log n)}$, $D = (R+2)/(k^2-1)$, $f = 2k$, $N' = 2^{R+Df} \leq 2^{2R} \leq 2^{2\,(\log N)^{1+3\beta/4+\beta/4}} \leq 2^{(\log n)^{O(1)}}$. Clearly, no polynomial time algorithm can distinguish whether the graph G with N' vertices has maximum clique size at least $c^D 2^R/2$ or at most 2^r unless NP $\subseteq$ BPTIME($2^{(\log n)^{O(1)}}$). Thus under this complexity assumption, no polynomial time algorithm can approximate MaxClique in a graph with N' vertices within factor N'^{α} where

$$\alpha = \frac{\log(c^D 2^R/2) - \log(2^r)}{\log N'} \geq \frac{R - 1 - \frac{R+2}{k^2-1} - r}{R + \frac{(R+2)2k}{k^2-1}} = 1 - \frac{\frac{(R+2)2k}{k^2-1} + 1 + r + \frac{R+2}{k^2-1}}{R + \frac{(R+2)2k}{k^2-1}} \geq$$

$$1 - \frac{\frac{(R+2)2k}{k^2-1} + 1 + r + \frac{R+2}{k^2-1}}{R} \geq 1 - O(\frac{r}{R}) \geq 1 - O(\frac{1}{(\log N)^{\beta/4}}) \geq 1 - \frac{1}{(\log N')^{\gamma}}$$

for some $\gamma > 0$. This proves Theorem 10.1.2 assuming NP $\not\subseteq$ BPTIME($2^{(\log n)^{O(1)}}$). One can use techniques from [33, Section 6.2, Theorem 6.16] to get the same result under the assumption NP $\not\subseteq$ ZPTIME($2^{(\log n)^{O(1)}}$). But this improvement is minor and we omit the details.

10.5 Randomized PCPs and Chromatic Number

Feige and Kilian [40] introduced the idea of *Randomized PCPs* to prove their hardness result for approximating chromatic number of a graph. We apply their technique to the PCP system constructed in Section 10.3 to obtain an improved result. Randomizing this system is easier than randomizing earlier PCP systems based on Long codes. First we give some basic definitions, the first one just restates Definition 3.4.1.

Definition 10.5.1 *An accepting pattern τ for a PCP verifier is a pair $\tau = (S, \nu)$ such that for some choice of the random string, S is the set of query bits read by the verifier and ν is a setting of these bits for which the verifier would accept. The set of all accepting patterns is denoted by $\mathcal{T}$. A proof Π is said to be consistent with a pattern $\tau = (S, \nu)$ if the values of bits in proof Π corresponding to the set S match ν.*

Definition 10.5.2 *A language L has a randomized PCP system with parameters (r, f, ρ, s), if there is a probabilistic polynomial time verifier V that can check membership proofs Π for language L using r random bits, f free query bits and satisfies :*

- *Soundness condition : $x \notin L \Longrightarrow \forall$ proofs Π, $\Pr\,[\, V$ accepts $\Pi\,] \le s$*
- *Covering condition : If $x \in L$, there exists a collection of proofs $\{\Pi_1, \Pi_2, \dots\}$ with a probability distribution on these proofs such that*

$$\forall\ \tau\ \in \mathcal{T},\ \ \Pr_i\,[\,\Pi_i \textit{ is consistent with } \tau\,]\ \ge \rho$$

The parameter ρ is called the covering parameter.

Covering condition basically says that one can have a distribution on proofs such that every accepting pattern is "covered" with probability ρ. In FGLSS graph G_0, this translates to a distribution on independent sets such that every vertex is covered with probability ρ. In other words, the *fractional chromatic number* $\chi_f(G_0)$ (and therefore the chromatic number $\chi(G_0)$) of the FGLSS graph is at most $1/\rho$.

Feige and Kilian [40] showed the following connection between randomized PCPs and hardness of approximating chromatic number of a graph. This theorem is obtained by taking inclusive graph product of the FGLSS graph and then taking a random induced subgraph of appropriate size. Since the statement of this theorem does not appear explicitly in their paper, we provide a brief sketch of the proof.

Theorem 10.5.3 *If there is RPCP system for 3SAT with parameters (r, f, ρ, s), then for any integer h, there is a randomized reduction from a 3SAT formula ϕ to a graph G' with $N' = (2^f/s)^h$ vertices such that : If ϕ is satisfiable, $\chi(G') \leq \frac{2\ln N'}{\rho^h}$ and if ϕ is unsatisfiable, with probability $1/2$, $\chi(G') \geq \frac{N'}{h2^{r+f}}$. This reduction runs in time polynomial in N' and the running time of the verifier.*

Proof: We will first apply Lemma 2 from [40]. The notations can be translated as $R \mapsto 2^r,\ e \mapsto s,\ \rho \mapsto \rho,\ A \mapsto 2^{r+f}$. This lemma reduces the RPCP system to the FGLSS graph G_0 that has 2^{r+f} vertices and satisfies :

- (Completeness/Covering condition) $\chi_f(G_0) \leq \frac{1}{\rho}$.
- (Soundness) $\alpha(G_0) \leq s2^r$.

Here $\alpha(G_0)$ is the size of maximum independent set in graph G_0 and $\chi_f(G_0)$ is the fractional chromatic number of G_0 (see [40, Definition 1]).

The final graph G' is obtained by taking the inclusive graph product ([40, Definition 2]) G_0^h and then randomly taking its vertex induced subgraph of size $N' = (\frac{|G_0|}{s2^r})^h = (\frac{2^f}{s})^h$. We apply Lemma 1 from [40] with the translation $G \mapsto G_0, G' \mapsto G', k \mapsto$

$h, C \mapsto s2^r$. Thus in the completeness case,

$$\begin{aligned}
\chi(G') &\leq (1+\ln|G'|)\,\chi_f(G') & [40, \text{Equation } 1] \\
&\leq (1+\ln|G'|)\,(\chi_f(G_0))^h & [40, \text{Lemma } 1] \\
&\leq \frac{2\ln N'}{\rho^h}
\end{aligned}$$

In the soundness case we have $\alpha(G_0) \leq s2^r$. Lemma 1 from [40] implies that with high probability $\alpha(G') \leq h|G_0|$. Hence

$$\chi(G') \geq \frac{|G'|}{\alpha(G')} \geq \frac{|G'|}{h|G_0|} = \frac{N'}{h2^{r+f}}$$

This completes the proof of Theorem 10.5.3. ■

We also need the following theorem which we prove in Section 10.7.

Theorem 10.5.4 *There exists an absolute constant $\mu < 1$ such that for any constant $\epsilon > 0$, there is a polynomial time reduction from a 3SAT formula ϕ with n variables to a regular Max-3-Lin-2 instance Γ with N variables such that if ϕ is unsatisfiable, at most μ fraction of equations in Γ can be satisfied and if ϕ is satisfiable, there exists a set of assignments $\mathcal{A} = \{\sigma_1, \sigma_2, \ldots\}$ for Γ such that every equation in Γ is satisfied by at least $(1-\epsilon)$ fraction of assignments in $\mathcal{A}$. We call Γ an instance of "coverable" Max-3-Lin-2(ϵ).*

Moreover this reduction can achieve $\epsilon = \frac{1}{(\log N)^\beta}$ for some constant $\beta > 0$ if we allow N and the running time of the reduction to be slightly superpolynomial i.e $n^{O(\log\log n)}$.

10.6 Randomized PCP for Coverable Max-3-Lin-2(ϵ)

A randomized PCP for coverable Max-3Lin(ϵ) is obtained by a simple modification of the verifier V_{lin} in Section 10.3.

The Randomization Technique : Let (P, Q) be the proofs provided to the Raz Verifier. We construct a new verifier V_{rand} as follows. The verifier V_{rand} has access to proofs $(\widehat{P}, \widehat{Q})$ where for some fixed l-bit string x, $\widehat{P}(v)$ is supposed to contain Hadamard code of the $(u + l)$-bit string $P(v) \circ x$ and $\widehat{Q}(w)$ is supposed to contain Hadamard code of the $(3u + l)$-bit string $Q(w) \circ x$ ($\circ$ denotes concatenation of strings). The string x acts as a dummy string. Recall that the covering condition in Definition 10.5.2 requires a collection of proofs instead of one single proof. The purpose of the dummy string x is precisely to generate different proofs. Different choices of the string x give different proofs. This idea is very similar to the *Randomized Label Cover* problem in Section 3.4.2.

If $\pi : \mathbb{F}_2^{3u} \to \mathbb{F}_2^{u}$ is the projection between w and v, we define a new projection function π' in the following way : $\pi' : \mathbb{F}_2^{3u+l} \to \mathbb{F}_2^{u+l}$ is defined by setting for every $\beta \in \mathbb{F}_2^{3u}$, $\eta \in \mathbb{F}_2^{l}$, $\pi'(\beta \circ \eta) = \pi(\beta) \circ \eta$.

The new verifier V_{rand} has access to proofs $(\widehat{P}, \widehat{Q})$ and it works in almost identical manner as the verifier V_{lin}. Its action is :

1. Pick sets $v, w_1, \ldots, w_k$ and corresponding projections $\pi_1, \ldots, \pi_k$. Construct new projection functions $\pi'_1, \ldots, \pi'_k$.

2. Let $A = \widehat{P}(v)$ and $B_j = \widehat{Q}(w_j)$ for $1 \leq j \leq k$.

3. Pick vectors $a_1, \ldots, a_k \in \mathbb{F}_2^{u+l}$ and $b_1, \ldots, b_k \in \mathbb{F}_2^{3u+l}$. Write them as $a_i = a'_i \circ a''_i$ and $b_i = b'_i \circ b''_i$ where a''_i and b''_i are l-bit vectors.

4. If $\{a''_i, b''_i : 1 \le i \le k\}$ are linearly dependent, then reject. Otherwise accept iff

$$A(a_i) \cdot B_j(b_j) = B_j(\pi_j'^{-1}(a_i) \oplus b_j) \; \forall \, i, j$$

Theorem 10.6.1 *The RPCP system with verifier V_{rand} for coverable Max-3-Lin-2(ϵ) instance Γ with N variables*

- *Uses $r = u \log N + O(ku)$ random bits and $2k$ free query bits.*
- *Covering parameter $\rho \ge 2^{-(2k+1)}$ provided $\epsilon k u \le 1/2$.*
- *Soundness $s \le 2^{-k^2+1}$ provided $u = \Omega(k^2)$.*

Proof: We note that $\{a''_i, b''_i\}$ is a collection of $2k$ vectors randomly chosen from a space of dimension l. We take $l = u \gg k$ and the probability that they are linearly dependent is negligible. So henceforth we ignore this issue.

Since V_{rand} works in a similar manner as V_{lin}, the soundness analysis for V_{lin} can be applied. Therefore the soundness of V_{rand} is bounded by $2^{-k^2} + \delta$ provided $C^u_{lin} < \delta^2$. Taking $\delta = 2^{-k^2}$ and $u = \Omega(k^2)$ ensures that the soundness is at most 2^{-k^2+1}.

Now we prove that this PCP system has good covering properties. For a (global) assignment σ to the instance Γ, let $\sigma(v)$ denote the assignment to variables in set v under σ. We define proofs $(\widehat{P}_{\sigma,x}, \widehat{Q}_{\sigma,x})$ corresponding to an assignment σ and a fixed l-bit string x where $\widehat{P}_{\sigma,x}(v)$ contains Hadamard code of the $(u + l)$-bit string $\sigma(v) \circ x$ and $\widehat{Q}_{\sigma,x}(w)$ contains Hadamard code of the $(3u + l)$-bit string $\sigma(w) \circ x$. Consider the collection of proofs

$$\{ \, (\widehat{P}_{\sigma,x}, \; \widehat{Q}_{\sigma,x}) \mid \; x \in \{0,1\}^l, \; \sigma \in \mathcal{A}\} \tag{10.4}$$

where $\mathcal{A}$ is a set of assignments such that every equation in Γ is satisfied by at least $1-\epsilon$ fraction of assignments in $\mathcal{A}$ (see Theorem 10.5.4). We consider the uniform probability distribution on these proofs.

Consider any pattern τ for the verifier V_{rand}. This pattern corresponds to some fixed setting of the bits $(A(a_1), \ldots, A(a_k), B_1(b_1), \ldots, B_k(b_k))$ where $A = Hadamard(\sigma(v) \circ x), B_j = Hadamard(\sigma(w_j) \circ x)$. By definition of the Hadamard codes, the values of these bits in the proof $(\widehat{P}_{\sigma,x}, \widehat{Q}_{\sigma,x})$ are (in {0,1} notation)

$$(a_1' \cdot \sigma(v) \oplus a_1'' \cdot x, \ldots, a_k' \cdot \sigma(v) \oplus a_k'' \cdot x,\ \ b_1' \cdot \sigma(w_1) \oplus b_1'' \cdot x, \ldots, b_k' \cdot \sigma(w_k) \oplus b_k'' \cdot x) \tag{10.5}$$

Since $\{a_i'', b_i'' : 1 \le i \le k\}$ are linearly independent, if x is a random string from $\{0,1\}^l$, the bit pattern (10.5) matches every $2k$ bit string with probability $1/2^{2k}$. On the other hand, if $\sigma \in \mathcal{A}$ is chosen randomly, with probability $1 - \epsilon k u$, all equations in $(w_j)_{j=1}^k$ will be satisfied. It follows that a randomly chosen proof from (10.4) is consistent with the pattern τ with probability $\ge (1-\epsilon k u)/2^{2k} \ge 1/2^{2k+1}$. This shows that the RPCP system has a good covering parameter. ■

10.6.1 Improved Inapproximability Result for Chromatic Number

We construct a RPCP system for 3SAT as follows. Using Theorem 10.5.4, we transform a 3SAT instance ϕ to a coverable Max-3-Lin-2(ϵ) instance Γ with $N = n^{O(\log\log n)}$ variables and $\epsilon = \frac{1}{(\log N)^\beta}$. Using Theorem 10.6.1, a RPCP system with the following parameters is constructed.

- $u = (\log N)^{3\beta/4},\ k = (\log N)^{\beta/4}$
- $r \le (\log N)^{1+3\beta/4}$

- Covering parameter $\rho \geq 2^{-(2k+1)}$ and the number of free bits $f = 2k$
- Soundness $s \leq 2^{-k^2+1}$

Note that the running time of the verifier and N is slightly superpolynomial. Now we apply the reduction given by Theorem 10.5.3 with $h = (\log N)^{1+\beta}$. The size of the graph produced is $N' = (2^f/s)^h = 2^{h(2k+k^2-1)} \leq 2^{(\log n)^{O(1)}}$. The gap in the chromatic number is $\frac{N'\rho^h}{h2^{r+f}\cdot 2\ln N'}$ which can be expressed as N'^{α} where

$$\alpha = \frac{\log N' - h\log(\frac{1}{\rho}) - \log h - r - f - 1 - \log(\ln N')}{\log N'} \geq 1 - O(\frac{h\log(\frac{1}{\rho})}{\log N'})$$

$$\geq 1 - O(\frac{hk}{h(2k+k^2-1)}) \ \geq 1 - O(\frac{1}{k}) \ \geq 1 - \frac{1}{(\log N')^{\gamma}}$$

for some $\gamma > 0$. This proves Theorem 10.1.3 assuming NP $\not\subseteq$ coRTIME($2^{(\log n)^{O(1)}}$) which is equivalent to the assumption NP $\not\subseteq$ ZPTIME($2^{(\log n)^{O(1)}}$).

10.7 Proof of Theorem 10.5.4

We first sketch Håstad's reduction from Gap-3SAT to Max-3-Lin-2 (which appears in Chapter 2). Håstad's verifier, which we call V_{3bit}, has access to a Gap-3SAT instance ϕ. For every set v of u variables, the verifier expects the long code of $\sigma(v)$ and for every set w of u clauses, it expects a long code of $\sigma(w)$ where σ is some global satisfying assignment. Here $\sigma(v), \sigma(w)$ are bit-strings of length u and $3u$ respectively.

The action of V_{3bit} is :

1. Pick a set v of u variables at random and a set w of u clauses, clause i containing variable i for $1 \leq i \leq u$.

2. Pick functions $f : \{0,1\}^u \mapsto \{-1,1\}$ and $g : \{0,1\}^{3u} \mapsto \{-1,1\}$ uniformly at random.

3. Pick an *error function* $\mu : \{0,1\}^{3u} \mapsto \{-1,1\}$ where for every $y \in \{0,1\}^{3u}$, $\mu(y)$ is set to 1 with probability $1-\epsilon'$ and $\mu(y)$ is set to -1 with probability ϵ'.

4. Let A be the supposed long code of $\sigma(v)$ in the proof and B be the supposed long code of $\sigma(w)$ in the proof. Let $\pi : \{0,1\}^{3u} \mapsto \{0,1\}^u$ be the projection function that restricts assignments to w to assignments to v.

5. Define $h : \{0,1\}^{3u} \mapsto \{-1,1\}$ as

$$h(y) = g(y) \cdot f(\pi(y)) \cdot \mu(y) \quad \forall\, y \in \{0,1\}^{3u}$$

The verifier accepts if and only if

$$A(f)B(g)B(fg\mu) = 1 \tag{10.6}$$

We state Håstad's result in a form convenient to us :

Theorem 10.7.1 *The verifier V_{3bit} has completeness $1-\epsilon'$ and soundness 0.6 provided $u \geq C_0 \log(1/\epsilon')$ where C_0 is some large absolute constant. The proof size is at most $n^{O(u)}2^{2^{3u}}$ where n is the size of the Gap-3SAT instance.*

Remark : The long codes are supposed to be *folded over true* and *conditioned upon 3SAT predicates*, but this is irrelevant for our purpose.

We note that if a proof contains correct long codes and is consistent with a satisfying assignment σ to Gap-3SAT formula ϕ, then equation (10.6) is satisfied if and only if $\mu(\sigma(w)) = 1$.

This reduction produces a weighted instance Γ of Max-3-Lin-2 where the variables correspond to bits in the proof and there is one equation of the form (10.6) corresponding to every tuple (A, B, f, g, μ) chosen by the verifier. (equation (10.6) is a linear equation modulo 2 in $\{0, 1\}$ notation). The equation corresponding to the tuple (A, B, f, g, μ) has weight equal to the probability with which this tuple is picked by the verifier. For any assignment to the variables in Γ, the weight of the equations satisfied by this assignment is equal to the probability with which the corresponding proof is accepted by the verifier. The size N of the instance Γ is polynomial in the proof size, i.e. $N \leq n^{C_1 u} 2^{2^{4u}}$ for some constant C_1.

Proof: (Of Theorem 10.5.4) We modify the verifier V_{3bit} using an idea similar to the randomization technique used in section 10.6. The new verifier expects the proof to contain long codes of strings $\sigma(v) \circ x$ and $\sigma(w) \circ x$ for some fixed l-bit string x. The projection function is modified accordingly. The verifier proceeds in a similar way, but now we have $f : \{0, 1\}^{u+l} \to \{-1, 1\}$ and $g, \mu, h : \{0, 1\}^{3u+l} \to \{-1, 1\}$.

We modify the instance Γ by deleting all equations corresponding to tuples (A, B, f, g, μ) where μ fails to satisfy the following condition :

$$\forall\, y \in \{0, 1\}^{3u},\ \mathrm{Pr}_{z \in \{0,1\}^l} \left[\, \mu(y \circ z) = -1 \,\right] \ \leq\ 2\epsilon' \tag{10.7}$$

For every $y \in \{0, 1\}^{3u}$, there are 2^l strings of the form $y \circ z$. We will set $\mu(y \circ z) = -1$ with probability ϵ' independently for all $y \circ z$. Using Chernoff bound, we can show that if $l = u = \Omega(\log(1/\epsilon'))$, the probability that an error function μ does not satisfy (10.7) is negligible compared to ϵ'. Thus the weight of the deleted equations is negligible and we ignore it from the analysis. Håstad's soundness analysis also applies to this new verifier implying that if ϕ is unsatisfiable, the maximum weight of the equations satisfied is at most 0.6.

For the covering condition, we consider the assignments σ_x to Γ given by proofs corresponding to a fixed satisfying assignment σ for ϕ and a choice of l-bit string x. As x ranges over all l-bit strings, we get different assignments to Γ.

An equation corresponding to the tuple (A, B, f, g, μ), where B is the long code of $\sigma(w) \circ x$, is satisfied provided $\mu(\sigma(w) \circ x) = 1$. This happens for at least $1 - 2\epsilon'$ fraction of x's since μ satisfies condition (10.7). Thus every equation is satisfied by at least $1 - 2\epsilon'$ fraction of assignments in the set $\{\sigma_x \mid x \in \{0,1\}^l\}$.

Taking $\epsilon = 2\epsilon'$ and transforming Γ to a *regular* Max-3-Lin-2 instance proves Theorem 10.5.4. The theorem also holds with a subconstant value of ϵ provided we allow the reduction to run in superpolynomial time. We take $u = C_2 \log\log n$ for a suitable constant C_2 so that $n^{C_1 u} = 2^{2^{4u}}$. With this choice we have $N \leq n^{C_1 u} 2^{2^{4u}} = 2^{2 \cdot 2^{4u}}$ and thus $\log N \leq 2 \cdot 2^{4u}$. By Theorem 10.7.1, we can achieve $\epsilon' = 2^{-u/C_0} \leq \frac{1}{(\log N)^{\beta}}$ for some $\beta > 0$. We also note that $N \leq n^{2 \cdot C_1 u} = n^{O(\log\log n)}$.

■

Chapter 11

Conclusion

More than decade's work by several researchers has resulted in the beautiful theory of inapproximability. Optimal hardness results are now known for many fundamental problems. However many important problems still remain open. In this chapter, we list some of them and point out some directions for future research.

11.1 Open Problems

These open problems are listed without any specific order (and with personal bias towards their importance). A more comprehensive list appears in Vazirani's book [113, Chapter 30].

1. **Vertex Cover** : Show $2-\epsilon$ hardness. The best known hardness is 1.36 [32]. How about vertex cover on k-uniform hypergraphs, is factor $k-\epsilon$ hard ? Factor $k-1-\epsilon$ hardness is known [29].

2. **Graph Coloring** : Show that 3-colorable graphs are hard to color with constantly many colors. How about $O(\log n)$ colors ? Currently, we know that 3-colorable

graphs can be colored (in poly-time) with $\tilde{O}(n^{3/14})$ colors [18] and it is NP-hard to color them with 4 colors [72].

3. **Independent Set** : In our opinion, before making any progress on vertex cover and graph coloring, it is important to show the following result : Show that there exists a constant α such that for any $\delta > 0$, it is NP-hard to find an independent set of (relative) size δ in a graph that is guaranteed to contain an independent set of size α. Showing $2 - \epsilon$ hardness for vertex cover means proving this result with $\alpha = \frac{1}{2} - \epsilon$. Such a result is equivalent to constructing a PCP with zero free bits, completeness α and soundness at most δ as shown in [16].

4. **Min-2SAT-Deletion** : Show any constant factor hardness. Currently, a constant factor hardness is known (that follows from Håstad's corresponding result for MAX-2SAT) and $O(\log n \log\log n)$ approximation is known [80].

5. **Graph Min Bisection, Sparsest Cut, Densest Subgraph, Bipartite Clique** : Show that these problems do not have a PTAS (unless of course P = NP). The best known approximation algorithms for these problems achieve ratios $O(\log^2 n)$ (see [42]), $O(\log n)$ (see [86]), $O(n^{1/3})$ (see [41]) and $O(n^{1/2})$ (folklore) respectively.

6. **Asymmetric TSP** : It has $O(\log n)$ approximation [45] and $\frac{117}{116} - \epsilon$ hardness [100]. A $4/3$ integrality gap example is known for a natural LP.

7. **Shortest Vector Problem in L_2 norm** : It has $2^{o(n)}$ approximation ([87], [107]) and $\sqrt{2} - \epsilon$ hardness [96]. Show any constant factor hardness. Does an approximation within polynomial factor exist ? Show any constant factor hardness in L_p norm for some fixed value of $1 \leq p < \infty$.

8. **Max Acyclic Subgraph** : It has $\frac{66}{65} - \epsilon$ hardness [99] and no algorithm better than trivial 2-approximation is known.

9. **Edge Disjoint Paths (Network Congestion Minimization)** : Approximation within ratio $O(\frac{\log n}{\log \log n})$ is known [102] and a matching integrality gap example is known (attributed to Leighton). A hardness factor 2 is known (it just follows from hardness of finding two edge disjoint paths between two pairs of terminals). Show any constant factor hardness.

10. **Bin-Packing** : Is it hard to find a bin-packing using $OPT + 1$ bins ? How about any additive constant ? The best algorithm uses $OPT + \log^2(OPT)$ bins [71].

It would be interesting to find new inapproximability thresholds for natural problems. Recently two such tight thresholds were shown : $\Omega(\log^2 n)$ for Group Steiner Tree on trees [57] and $\Omega(\log^* n)$ for Asymmetric k-Center [26]. Apart from showing hardness results for specific problems, it would be nice to relate approximability of different problems and get some reasonable classification.

11.2 Future Directions

Current techniques seem to have reached their limits for problems like Vertex Cover and Graph Min Bisection. Recently, two new approaches have been proposed. The first is the Unique Games Conjecture presented in this thesis and the second is Feige's hypothesis about hardness of Random 3SAT.

11.2.1 Unique Games Conjecture

As seen in this thesis, this conjecture implies $2 - \epsilon$ hardness for Vertex Cover and any constant factor hardness for Min-2SAT-Deletion. Assuming a stronger form of this conjecture (with additional assumption that the underlying bipartite graph of unique 2-prover game is an expander), one can show any constant factor hardness for Graph Min Bisection.

11.2.2 Feige's Hypothesis about Random 3SAT

A random 3SAT formula with density C is obtained by picking $m = Cn$ clauses uniformly at random from the set of all possible clauses on n variables. A ***refuting procedure*** is a procedure that says YES on satisfiable 3SAT formulae and says YES/NO on unsatisfiable instances. Note that for large C, almost every 3SAT formula is unsatisfiable.

Feige's Hypothesis : For all large constants C, there is no poly-time refuting procedure that says NO on a constant fraction (say 50%) of unsatisfiable 3SAT formulae picked with density C.

Feige [34] shows that this hypothesis implies that there is no PTAS for Graph Min Bisection, Bipartite Clique, Densest Subgraph and Catalog Segmentation. The idea is that when a 3SAT formula is picked at random, then an appropriate reduction to a graph problem will give a random-looking graph (and hence a graph with expansion properties). Following Feige's paper, Alekhnovich [5] has shown interesting results based on (conjectured) hardness of random system of linear equations.

Bibliography

[1] M. Ajtai. Generating hard instances of lattice problems. In *Proc. 28th ACM Symposium on the Theory of Computing*, pages 99–108, 1996.

[2] M. Ajtai. The shortest vector problem in L_2 is NP-hard for randomized reductions. In *Proc. 30th ACM Symposium on the Theory of Computing*, pages 10–19, 1998.

[3] M. Ajtai and C. Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *Proc. 29th ACM Symposium on the Theory of Computing*, pages 284–293, 1997.

[4] M. Ajtai, R. Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *Proc. of the 33rd ACM Symposium on the Theory of Computing*, pages 601–610, 2001.

[5] M. Alekhnoivh. More on average case vs approximation complexity. In *Proc. 44th IEEE Symposium on Foundations of Computer Science*, 2003.

[6] N. Alon and N. Kahale. Approximating the independence number via the θ-function. *Technical Report, Tel Aviv University*, 1995.

[7] G. Andersson, L. Engebretsen, and J. Hastad. A new way of using semidefinite programming with applications to linear equations mod p. *Journal of Algorithms*, 39(2):162–204, 2001.

[8] S. Arora. *Probabilistic checking of proofs and the hardness of approximation problems*. Ph.D. thesis, UC Berkeley, 1994.

[9] S. Arora, L. Babai, J. Stern, and E. Sweedyk. The hardness of approximate optima in lattices, codes and systems of linear equations. *Journal of Computer and Systems Sciences*, 54:317–331, 1997.

[10] S. Arora, B. Bollobas, and L. Lovász. Proving integrality gaps without knowing the linear program. In *Proc. 43rd IEEE Foundations of Computer Science*, 2002.

[11] S. Arora and C. Lund. *Approximation Algorithms for NP-hard Problems, editor : D. Hochbaum*. PWS Publishing, 1996.

[12] S. Arora, C. Lund, R. Motawani, M. Sudan, and M. Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, 1998.

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

[14] L. Babai. Trading group theory for randomness. In *Proc. 17th ACM Symposium on Theory of Computing*, pages 421–429, 1985.

[15] L. Babai, L. Fortnow, and C. Lund. Non-deterministic exponential time has two-prover interactive protocols. pages 113–121, 1988.

[16] M. Bellare, O. Goldreich, and M. Sudan. Free bits, PCPs and non-approximability. *Electronic Colloquium on Computational Complexity, Technical Report TR95-024*, 1995.

[17] M. Bellare and M. Sudan. Improved non-approximability results. In *Proc. 26th ACM Symposium on Theory of Computing*, pages 184–193, 1994.

[18] A. Blum and D. Karger. An $\tilde{O}(n^{3/14})$ coloring algorithm for 3-colorable graphs. *Information Processing Letters*, 61:49–53, 1997.

[19] M. Blum and S. Kannan. Designing programs that check their work. In *Proc. 21st ACM Symposium on Theory of Computing*, pages 86–97, 1989.

[20] M. Blum, M. Luby, and R. Rubinfeld. Self-testing/correcting with applications to numerical problems. In *Proc. 22nd ACM Symposium on Theory of Computing*, pages 73–83, 1990.

[21] R. Boppana and M. Halldórsson. Approximating maximum independent sets by excluding subgraphs. In J. R. Gilbert and R. Karlsson, editors, *Proc. 92nd Scandinavian Workshop on Algorithm Theory*, volume 447, pages 13–25, 1990.

[22] J. Bourgain. On the distribution of the Fourier spectrum of boolean functions. *manuscript*.

[23] J. Cai. Applications of a new transference theorem to Ajtai's connection factor. In *Proc. 14th IEEE Conference on Computational Complexity*, 1999.

[24] J. Cai and A. Nerurkar. An improved worst-case to average-case connection for lattice problems. In *Proc. 38th IEEE Symposium on Foundations of Computer Science*, 1997.

[25] J. Cai and A. Nerurkar. Approximating the SVP to within a factor $(1 + 1/\dim^{\epsilon})$ is NP-hard under randomized reductions. In *Proc. 13th IEEE Conference on Computational Complexity*, pages 151–158, 1998.

[26] J. Chuzhoy, S. Guha, E. Halperin, S. Khanna, G. Kortsarz, R. Krauthgamer, and S. Naor. Asymmetric k-center is $\log^* n$-hard to approximate. *Personal communication*.

[27] I. Dinur. Approximating SVP$_\infty$ to within almost polynomial factors is NP-hard. In *Proc. 4th Italian Conference on Algorithms and Complexity*, volume LNCS : 1767. Springer, 2000.

[28] I. Dinur, V. Guruswami, and S. Khot. Vertex cover on k-uniform hypergraphs is hard to approximate within factor $(k - 3 - \epsilon)$. *Electronic Colloquium on Computational Complexity, Technical Report TR02-027*, 2002.

[29] I. Dinur, V. Guruswami, S. Khot, and O. Regev. A new multilayered PCP and the hardness of hypergraph vertex cover. In *Proc. 34thth ACM Symposium on Theory of Computing*, 2002.

[30] I. Dinur, G. Kindler, and S. Safra. Approximating CVP to within almost-polynomial factors is NP-hard. In *Proc. 39th IEEE Symposium on Foundations of Computer Science*, 1998.

[31] I. Dinur, O. Regev, and C. Smyth. The hardness of 3-uniform hypergraph coloring. In *Proc. 43rd IEEE Symposium on Foundations of Computer Science*, 2002.

[32] I. Dinur and S. Safra. The importance of being biased. In *Proc. 34th Annual ACM Symposium on Theory of Computing*, 2002.

[33] L. Engebretsen and J. Holmerin. Towards optimal lower bounds for clique and chromatic number. *Electronic Colloquium on Computational Complexity (ECCC)*, (TR01-003), 2001.

[34] U. Feige. Relations between average case complexity and approximation complexity. In *Proc. 34th ACM Symposium on Theory of Computing*.

[35] U. Feige. Error reduction - the state of the art. *Technical Report CS95-32, Weizmann Institute of Technology*, 1995.

[36] U. Feige. Randomized graph products, chromatic numbers, and the lovász θ-function. In *Proc. 27th ACM Symposium on Theory of Computing*, pages 635–640, 1995.

[37] U. Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.

[38] U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy. Interactive proofs and the hardness of approximating cliques. *Journal of the ACM*, 43(2):268–292, 1996.

[39] U. Feige, M. Halldorsson, and G. Kortsarz. Approximating the domatic number. In *Proc. 32nd ACM Symposium on Theory of Computing*, pages 134–143, 2000.

[40] U. Feige and J. Kilian. Zero knowledge and the chromatic number. In *Proc. 11thIEEE Conference on Computational Complexity*, pages 278–287, 1996.

[41] U. Feige, G. Kortsarz, and D. Peleg. The dense k-subgraph problem. *Algorithmica*, 29(3):410–421, 2001.

[42] U. Feige and R. Krauthgamer. A polylogarithmic approximation of the minimum bisection. In *Proc. 41st IEEE Symposium on Foundations of Computer Science*, pages 105–115, 2000.

[43] U. Feige and L. Lovász. Two-prover one-round proof systems, their power and their problems. In *Proc. 24th Annual ACM Symposium on Theory of Computing*, pages 733–744, 1992.

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

[45] A. Frieze, G. Galbiati, and F. Maffioli. On the worst-case performance of some algorithms for the asymmetric traveling salesman problem. *Networks*, 12:23–39, 1982.

[46] A. Frieze and M. Jerrum. Improved approximation algorithms for MAX k-CUT and MAX BISECTION. *Algorihmica*, 18:67–81, 1997.

[47] M. Fürer. Improved hardness results for approximating the chromatic number. In *Proc. 36th IEEE Sumposium on Foundations of Computer Science*, pages 414–421, 1995.

[48] M. R. Garey and D. S. Johnson. The complexity of near-optimal graph coloring. *Journal of the ACM*, 23:43–49, 1976.

[49] C. Gauss. Disquisitiones arithmeticae (leipzig, 1801 : art. 171). Yale Univ. Press. English translation by A.A. Clarke, 1966.

[50] M. Goemans and D. Williamson. 0.878 approximation algorithms for MAX-CUT and MAX-2SAT. In *Proc. 26th ACM Symposium on Theory of Computing*, pages 422–431, 1994.

[51] O. Goldreich. Using the FGLSS-reduction to prove inapproximability results for minimum vertex cover in hypergraphs. *Electronic Colloquium on Computational Complexity, Technical Report TR01-102*, 2001.

[52] O. Goldreich and S. Goldwasser. On the limits of non-approximability of lattice problems. In *Proc. 30th ACM Symposium on the Theory of Computing*, pages 1–9, 1998.

[53] S. Goldwasser, S. MIcali, and C. Rackoff. The knowledge complexity of interactive proofs. *SIAM Journal on Computing*, 18:186–208, 1989.

[54] R. L. Graham, M. Grötschel, and L. Lovász, editors. *Handbook of combinatorics*, volume 1, 2. Elsevier Science B.V., Amsterdam, 1995.

[55] V. Guruswami, J. Hastad, and M. Sudan. Hardness of approximate hypergraph coloring. In *Proc. 41st IEEE Symposium on Foundations of Computer Science*, pages 149–158, 2000.

[56] E. Halperin. Improved approximation algorithms for the vertex cover problem in graphs and hypergraphs. In *Proc. 11th ACM-SIAM Symposium on Discrete Algorithms*, pages 329–337, 2000.

[57] E. Halperin and R. Krauthgamer. Polylogarithmic inapproximability. In *Proc. 35th ACM Symposium on Theory of Computing*, 2003.

[58] E. Halperin, R. Nathaniel, and U. Zwick. Coloring k-colorable graphs using smaller palettes. In *Proc. 12th ACM-SIAM Symposium on Discrete Algorithms*, pages 319–326, 2001.

[59] J. Hastad. Clique is hard to approximate within $n^{1-\epsilon}$. In *Proc. 37th IEEE Symposium on Foundations of Computer Science*, pages 627–636, 1996.

[60] J. Hastad. Some optimal inapproximability results. In *Proc. 29th ACM Symposium on Theory of Computing*, pages 1–10, 1997.

[61] J. Hastad. On a protocol possibly useful for MIN-2SAT. *manuscript*, 2001.

[62] J. Hastad and S. Khot. Query efficient PCPs with perfect completeness. In *Proc. 42nd IEEE Symposium on Foundations of Computer Science*, 2001.

[63] J. Hastad and V. Srinivasan. On the advantage over a random assignment. In *Proc. 34th ACM Symposium on Theory of Computing*, 2002.

[64] J. Hastad and A. Wigderson. Simple analysis of graph tests for linearity and PCP. In *Proc. 16th IEEE Conference on Computational Complexity*, 2001.

[65] J. Holmerin. Improved inapproximability results for vertex cover on k-uniform hypergraphs. In *Proc. 29th International Colloquium on Automata, Languages and Programming*, pages 1005–1016, 2002.

[66] J. Holmerin. Vertex cover on 4-regular hyper-graphs is hard to approximate within $2 - \epsilon$. In *Proc. 34th ACM Symposium on Theory of Computing*, 2002.

[67] D. S. Johnson. Approximation algorithms for combinatorial problems. *Journal of Computer and System Sciences*, 9:256–278, 1974.

[68] R. Kannan. Improved algorithms for integer programming and related lattice problems. In *Proc. 15th ACM Symposium on Theory of Computing*, pages 193–206, 1983.

[69] R. Kannan. Minkowski's convex body theorem and integer programming. *Mathematics of Operations Research*, 12:415–440, 1987.

[70] D. Karger, R. Motwani, and M. Sudan. Approximate graph coloring by semidefinite programming. In *Proc. 35th IEEE Symposium on Foundations of Computer Science*, pages 2–13, 1994.

[71] N. Karmarkar and R. Karp. An efficient approximation scheme for the one-dimensional bin packing problem. In *Proc. 23rd IEEE Symposium on Foundations of Computer Science*, pages 312–320, 1982.

[72] S. Khanna, N. Linial, and S. Safra. On the hardness of approximating the chromatic number. In *Proc. 2nd Israel Symposium on Theory and Computing Systems, ISTCS*, pages 250–260, 1993.

[73] S. Khanna, R. Motwani, M. Sudan, and U. Vazirani. On syntactic versus computational views of approximability. In *Proc. 35th IEEE Symposium on Foundations of Computer Science*, pages 819–830, 1994.

[74] S. Khot. Improved inapproximability results for maxclique, chromatic number and approximate graph coloring. In *Proc. 42nd IEEE Annual Symposium on Foundations of Computer Science*, 2001.

[75] S. Khot. Hardness of coloring 3-colorable 3-uniform hypergraphs. In *Proc. 43rd IEEE Symposium on Foundations of Computer Science*, 2002.

[76] S. Khot. Hardness rsults for approximate hypergraph coloring. In *Proc. 34th ACM Symposium on Theory of Computing*, 2002.

[77] S. Khot. On the power of unique 2-prover 1-round games. In *Proc. 34th ACM Symposium on Theory of Computing*, 2002.

[78] S. Khot. Hardness of approximating the shortest vector problem in high L_p norms. In *Proc. 44th IEEE Symposium on Foundations of Computer Science*, 2003.

[79] S. Khot and O. Regev. Vertex cover might be hard to approximate to within $2 - \epsilon$. In *Proc. 18th IEEE Conference on Computational Complexity*, 2003.

[80] P. Klein, S. Plotkin, S. Rao, and E. Tardos. Approximation algorithms for steiner and directed multicuts. *Journal of Algorithms*, 22(2):241–269, 1997.

[81] M. Krivelevich, R. Nathaniel, and B. Sudakov. Approximating coloring and maximum independent set in 3-uniform hypergraphs. In *Proc. 12th ACM-SIAM Symposium on Discrete Algorithms*, 2001.

[82] R. Kumar and D. Sivakumar. Complexity of SVP - A reader's digest. In L. Hemaspaandra, editor, *SIGACT News, Complexity Theory Column*, volume 32(3). 2001.

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

[84] J. Lagarias and A. Odlyzko. Solving low-density subset sum problems. *Journal of the ACM*, 32(1):229–246, 1985.

[85] S. Landau and G. Miller. Solvability of radicals is in polynomial time. *Journal of Computer and Systems Sciences*, 30(2):179–208, 1985.

[86] T. Leighton and S. Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *Journal of the ACM*, 46:787–832, 1999.

[87] A. Lenstra, H. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Ann.*, 261:513–534, 1982.

[88] H. Lenstra. Integer programming with a fixed number of variables. *Tech. Report 81-03, Univ. of Amsterdam*, 1981.

[89] R. Lipton. Efficient checking of computations. In *Proc. 6th Symposium on Theoretical Aspects of Computer Science*, 1989.

[90] L. Lovász. Coverings and colorings of hypergraphs. In *Proc. 4th Southeastern Conf. on Combinatorics, Graph Theory and Computing*, pages 3–12. Utilitas Mathematica Publishing, Winnipeg, 1973.

[91] L. Lovász. On the ratio of optimal integral and fractional covers. *Discrete Mathematics*, 13:383–390, 1975.

[92] C. Lund, L. Fortnow, H. Karloff, and N. Nisan. Algebraic methods for interactive proof systems. *Journal of the ACM*, 39(4):859–868, 1992.

[93] C. Lund and M. Yannakakis. On the hardness of approximating minimization problems. *Journal of the ACM*, 41:960–981, 1999.

[94] G. Margulis. Probabilistic characteristics of graphs with large connectivity. *Problemy Peredaci Informacii*, 10(2):101–108, 1974.

[95] D. Micciancio. *On the hardness of the shortest vector problem*. PhD Thesis, MIT, 1998.

[96] D. Micciancio. The shortest vector problem is NP-hard to approximate to within some constant. In *Proc. 39th IEEE Symposium on Foundations of Computer Science*, 1998.

[97] D. Micciancio and S. Goldwasser. *Complexity of Lattice Problems, A Cryptographic Perspective*. Kluwer Academic Publishers, 2002.

[98] H. Minkowski. *Geometrie der zahlen*. Tuebner, 1910.

[99] A. Newman. *Approximating the maximum acyclic subgraph*. Masters' Thesis, MIT, 2000.

[100] C. Papadimitriou and S. Vempala. On the approximability of the traveling salesman problem. In *Proc. 32nd ACM Symposium on the Theory of Computing*, 2000.

[101] C. Papadimitriou and M. Yannakakis. Optimization, approximation, and complexity classes. *Journal of Computer and Systems Sciences*, 43:425–440, 1991.

[102] P. Raghavan and C. D. Thompson. Randomized rounding: a technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4).

[103] R. Raz. A parallel repetition theorem. *SIAM J. of Computing*, 27(3):763–803, 1998.

[104] O. Regev. New lattice based cryptographic constructions. In *Proc. 35th ACM Symposium on the Theory of Computing*, 2003.

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

[106] A. Samorodnitsky and L. Trevisan. A PCP characterization of NP with optimal amortized query complexity. In *Proc. 32nd ACM Symposium on Theory of Computing*, pages 191–199, 2000.

[107] C. Schnorr. A hierarchy of polynomial-time basis reduction algorithms. In *Proc. of Conference on Algorithms*, pages 375–386, 1985.

[108] A. Shamir. IP = PSPACE. *Journal of the ACM*, 39(4):869–877, 1992.

[109] A. Srinivasan. The value of strong inapproximability results for clique. In *Proc. 32nd ACM Symposium on Theory of Computing*, pages 144–152, 2000.

[110] L. Trevisan. Recycling queries in PCPs and in linearity tests. In *Proc. 30th ACM Symposium on Theory of Computing*, 1998.

[111] L. Trevisan. Non-approximability results for optimization problems on bounded degree instances. In *Proc. 33rd ACM Symposium on Theory of Computing*, pages 453–461, 2001.

[112] P. van Emde Boas. Another NP-complete problem and the complexity of computing short vectors in a lattice. *Tech. Report 81-04, Mathematische Instiut, Univ. of Amsterdam*, 1981.

[113] V. V. Vazirani. *Approximation Algorithms*. Springer, 2001.

[114] D. Zuckerman. On unapproximable versions of NP-complete problems. *SIAM J. on Computing*, pages 1293–1304, 1996.

[115] U. Zwick. Approximation algorithms for constraint satisfaction problems involving at most three variables per constraint. In *Proc. 9th ACM-SIAM Symposium on Discrete Algorithms*, pages 201–210, 1998.

[116] U. Zwick. Finding almost satisfying assignments. In *Proc. 30th ACM Symposium on Theory of Computing*, pages 551–560, 1998.