New Directions in Inapproximability: Promise Constraint Satisfaction Problems and Beyond

Sai Sandeep Reddy Pallerla

CMU-CS-22-139
August 2022

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Venkatesan Guruswami, Chair
Anupam Gupta
Ryan O'Donnell
Nikhil Bansal, University of Michigan

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

This research was sponsored by Google, The David and Lucille Packard Foundation under award number 200529094A, and the National Science Foundation under award numbers CCF-1422045, CCF-1563742, CCF-1814603, and CCF-1908125. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government or any other entity.

Keywords: Approximation Algorithms, Hardness of Approximation, Approximate Graph Coloring, Promise Constraint Satisfaction Problems, Polymorphisms, Hypergraph Rainbow Coloring, Semi Definite Programming Relaxations, Set Cover, Vector Bin Packing, Hypergraph Vertex Cover

To my parents and my sister

Abstract

The field of hardness of approximation has seen a lot of progress in the past three decades resulting in almost optimal inapproximability results for many computational problems, including all Constraint Satisfaction Problems(CSPs). However, our understanding of inapproximability is still rather limited for some fundamental problems such as approximate graph coloring, and problems for which approximation algorithms have been widely studied, for example, clustering, packing, and scheduling problems. In this thesis, we make progress on these questions by studying Promise CSPs that generalize CSPs, and more abstractly, computational problems with a given promise, either a solution satisfying a strong property, or a structural guarantee on the underlying instance.

Promise Constraint Satisfaction Problems (PCSPs) generalize the traditional CSPs by allowing for a weaker and stronger form for each predicate. PCSPs have received a lot of attention recently, both on the algorithmic and hardness front, and their study has led to breakthroughs in approximate graph and hypergraph coloring problems. In this thesis, we continue that line of work, obtaining new characterization of polynomial time solvability for several classes of Promise CSPs including Boolean monotone PCSPs, variants of graph and hypergraph colorings. We also study robust algorithms for Promise CSPs, and give a dichotomy result characterizing when certain classes of PCSPs have robust algorithms. More generally, we study combinatorial problems under a given structural promise – for example, set cover on set systems where every pair of sets intersect in at most one element. We use inapproximability results on such structured instances to resolve the approximability of multidimensional packing problems and scheduling with communication delays.

Acknowledgements

First and foremost, I feel deeply fortunate to work with Venkat. He has been a wonderful advisor who adapts to the students' needs, mentoring them closely at times, and giving them freedom when needed. I am thankful to him for all the countless hours spent working on research, his superhuman patience, and for being an inspiration, both as a researcher and a person. Apart from the technical aspects, I have learned many things working with him, including how to pick problems to work on and how to ask interesting and important questions—even during the talks, he asks simple yet deep questions every single time, which I hope I can emulate someday. He is a treasure trove of ideas, and irrespective of the project stage, be it the early brainstorming stage or when I am stuck on something or when we are working on a write-up, after meeting him (mostly virtually, unfortunately), I always end up with much better clarity and having many new ideas to work on.

I owe my gratitude to other members of my thesis committee: Anupam Gupta, Ryan O'Donnell, and Nikhil Bansal. Talking to Anupam, with his constant optimism, is always enlightening. I have benefited from his inspiring teaching style, by attending his lectures and by being a TA. I am especially grateful to him for going out of his way to give feedback on how to improve my presentation skills. I had great fun binge-watching Ryan's youtube lectures on topics ranging from analysis of Boolean functions to infinite expander graphs. Reading the lecture notes of a course taught by Ryan and Venkat at the University of Washington on PCPs during my undergrad was one of the earliest experiences that helped develop my interest in theoretical computer science, and I feel privileged to have both Ryan and Venkat on my thesis committee. I am also indebted to Anupam and Ryan for helping me with my Ph.D. requirements. I thank Nikhil for his encouragement and for his comments on an early draft that were greatly helpful. I am thankful to Pravesh Kothari with whom I enjoyed talking about things, research and otherwise, in the last semester. I am also grateful to Deb Cavlovich for all the help with administrative matters.

Most of the work in this thesis is a joint work and I am thankful to my collaborators: Joshua Brakensiek, Sami Davies, Mordecai Golin, Varun Gupta, Ravishankar Krishnaswamy, Janardhan Kulkarni, Amit Kumar, Euiwoong Lee, Jakub Opršal, Arka Ray, Thomas Rothvoss, Janani Sundaresan, Jakub Tarnawski, Yihao Zhang. While not every collaboration has led to concrete results, I have thoroughly enjoyed every discussion and have learned a lot. I also thank Libor Barto, Amey Bhangale, Boris Bukh, Vincent Cohen-Addad, Florian Frick, Abhishek Shetty, and Xinyu Wu for helpful discussions.

For the past few years, Pittsburgh has become a home away from home. Having lived in zero other cities in the US, I still claim that Pittsburgh is one of the best cities in the US, where you can get everything that a big city offers, without actually feeling like you're in one. But the reason it has become home is the people and I am grateful to the friends here in Pittsburgh: Adithya, without whom I am not sure how I would have survived the pandemic; other roommates (not all at once) Rawal, Eyan, Namit; Prashanth, who helped me during the early days when I moved to the US; Sakshi, who has been incredibly supportive and encouraging (and also introduced me to *The Marvelous Mrs. Maisel*); Abhiram, Andrii, Ankush, Anup, Biswa, Karthik, Leqi, Nirav, Roger, Shilpa, Sitoshna who have been part of my pandemic social bubble (at various points) and have been there for me through the highs and lows of grad school. I also had fun group lunch/dinners with Ainesh, Alperen, Jackson, Jeff, Joao, Jonathan, Madhusudan, Nic, Omar, Peter, Prashanti, Tim, Vijay; with the Microsoft Research gang at CMU including Anish, Chirags (Gupta and Pabbaraju), Don, Nithin, Phillip, Rahul, Raut, Suhas; Biking and tennis with Akarsh, Jatin, Mihir, Sol; Avalon with David, Guillaume, and Marissa.

Before CMU, I spent a wonderful year at Microsoft Research in Bangalore working with Ravishankar Krishnaswamy. I am grateful to Ravi for his continued encouragement and support during grad school. Even before that, during my undergrad, I am grateful to Ajit Diwan and Nutan Limaye for introducing me to discrete math and theoretical computer science research.

Finally, I am thankful to my family: my uncle and aunt in Virginia, who have always been there for me here in the US, and my parents, my sister Navya, and my cousin Tarak for their unconditional love and encouragement.

Contents

List of Figures

Chapter 1

Introduction

Starting with the celebrated PCP theorem [Aro+98], the study of the hardness of approximation has played a key role in theory of computing, including results such as Parallel Repetition [Raz98], and the study of Unique Games Conjecture (UGC) [Kho02b]. This has led to optimal inapproximability results for various computational problems such as 3-SAT [Hås01], and more generally, all Constraint Satisfaction Problems [Rag08] (modulo UGC). Despite this progress, our understanding of approximation algorithms for some problems is lacking, for example, approximate graph coloring, and problems that have been well studied in the algorithms community such as clustering, packing, and scheduling problems.

In this thesis, we make progress on these problems by studying computational problems *under a given promise*. There are two different kinds of promises on the instances: first, we are promised that the instance has a solution that satisfies a stronger property while the goal is to find a solution with a weaker property. For example, given a graph that is promised to be 3-colorable, can we color it with 6 colors in polynomial time? This is an example of Promise Constraint Satisfaction Problems (PCSPs) that generalize the classical Constraint Satisfaction Problems (CSPs) by having weak and strong predicate pairs and the goal is to find a solution satisfying the weak predicates under the promise that there is a solution satisfying the strong predicate. PCSPs are a vast generalization of CSPs, capturing problems such as approximate graph and hypergraph coloring, $(2+\epsilon)$-SAT [AGH17].

Formally introduced in a work of Austrin, Guruswami, and Hastad [AGH17], there has been a flurry of works on PCSPs, both on the algorithmic and hardness front. In this thesis, we continue this line of works. Regarding specific PCSPs, for the approximate graph and hypergraph coloring, we prove the hardness of $O(1)$-coloring a 3-colorable graphs [GS20a] under a weaker conjecture, namely the d-to-1 conjecture of Khot [Kho02b]. Furthermore, we prove improved hardness of rainbow coloring of hypergraphs [GS20b], and also apply these hardness results to prove almost optimal hardness results for Vector Scheduling and Vector Bin Covering [San21]. For the Boolean case, we prove a conditional dichotomy result for monotone Boolean PCSPs [BGS21]. We also study *robust* algorithms for PCSPs where the goal is to output a solution satisfying $1 - f(\epsilon)$ fraction of the constraints on instances promised to be $1-\epsilon$ satisfiable, with $f(\epsilon) \to 0$ as $\epsilon \to 0$.

A different way of studying computational problems under a promise is when the instances themselves have a strong structural property. As a concrete example, consider the set cover problem where given a set system, the objective is to find the minimum number of sets whose union is the whole universe of elements. There is $\ln n$-factor approximation algorithm for the problem, and this is tight [Fei98]. Suppose that the instance has stronger structural property, namely, that any two sets in the family intersect in at most one element i.e., the underlying set system is *simple* – can we get improved algorithms for the problem, or does the same hardness hold? It turns out that the set cover problem on simple set systems is a useful problem to study, with connections to several other problems. We prove a hardness result on the set cover problem on simple set systems and use it to show optimal (up to constants) hardness of approximation for Vector Bin Packing [San21], whose approximability has been open for more than two decades. The set cover problem on simple set systems is also closely related to the Tuza's conjecture relating to packing and covering of edges of a graph with triangles, and we use the algorithmic ideas used in the set cover problem to prove the fractional version of the generalized Tuza's conjecture [GS22]. Finally, we use the idea of studying computational problems under promise to resolve the approximability of scheduling with non-uniform communication delays [Dav+22], where we introduce a problem called Unique Machines Precedence constraints Scheduling (UMPS) which is a very structured instance of unrelated machine scheduling with precedence constraints problem.

1.1 Promise Constraint Satisfaction Problems (PCSPs)

As mentioned earlier, Promise Constraint Satisfaction Problems (PCSPs) are a vast generalization of CSPs, where each predicate now has a weak and strong form. The central question in the study of PCSPs is whether there exists a complexity dichotomy of CSPs extends to PCSPs i.e. if every PCSP is either in P or is NP-complete. We first give a historic overview of the CSP dichotomy theorem. The quest for CSP dichotomy started with a result of Schaefer who proved that every Boolean CSP is either in P or is NP-Hard [Sch78]. Feder and Vardi [FV98] conjectured that the same should hold over arbitrary domains as well. They also showed that the then known algorithmic results all follow by the algebraic closure properties of the CSPs.

This notion was formalized by Jeavons, Cohen, and Gyssens [JCG97; Jea98] and other works [BJK05] that crystallized the *(universal) algebraic approach* to CSPs. In the algebraic approach, the higher-order closure properties obeyed by the predicates, namely their *polymorphisms*, are studied. A polymorphism is a function that when applied coordinate-wise to arbitrary satisfying assignments to the predicate, is guaranteed to produce an output that satisfies the predicate. For example, consider an arbitrary instance I of the 2-SAT problem over n variables, and suppose that $\mathbf{x}, \mathbf{y}, \mathbf{z} \in \{0,1\}^n$ are three assignments that satisfy all the constraints in I. Now, if we compute $\mathbf{u} \in \{0,1\}^n$ that is obtained by setting $u_i = \mathsf{MAJ}(x_i, y_i, z_i)$ for all $i \in [n]$, the assignment $\mathbf{u}$ also satisfies all the constraints of I. Thus, the majority function on 3 bits is a polymorphism of the 2-SAT CSP. On the other hand, for the 3-SAT problem, it is not hard to prove that the only polymorphisms are the dictator functions. The algebraic approach has been immensely successful and culminated in the recent resolution of the Feder-Vardi conjecture by Bulatov [Bul17] and

Zhuk [Zhu20]. Further, these proofs yield a precise understanding of the mathematical structure underlying efficient algorithms: if the CSP has a "non-trivial" polymorphisms, the CSP is polytime solvable, and otherwise, it is NP-complete.

The algebraic approach is the key tool towards establishing such a dichotomy result even for PCSPs. The Galois correspondence from the CSP world extends to PCSPs, i.e., the polymorphisms fully capture the computational complexity of the underlying PCSP [Pip02a; BG21a]. This has been extended to show that just the identities satisfied by the polymorphisms suffice to capture the computational complexity of the underlying PCSP [Bar+21]. However, the polymorphisms of PCSPs are much richer, and characterizing which polymorphisms lead to algorithms and which ones lead to hardness has been a challenging problem. Conceptually, the principal difficulty is that the polymorphisms for CSPs are closed under composition (hence referred to as *clones*), whereas for PCSPs, this is no longer the case. As a result, even in the Boolean case, we do not have a dichotomy theorem for PCSPs. On the other hand, this richness of PCSPs has motivated a lot of recent works, both understanding fixed template PCSPs such as variants of graph and hypergraph colorings [Bar+21; KO19], variants of Boolean PCSPs [AGH17; BŽ21], understanding the power of various algorithmic techniques for PCSPs [Bra+20; CŽ22b].

In this thesis, we continue this line of works. In particular, we prove the following results regarding Promise CSPs.

Conditional Dichotomy of Boolean Monotone PCSPs. Towards establishing a potential Boolean PCSP dichotomy, progress has been made by Ficak, Kozik, Olsák, and Stankiewicz [Fic+19], who obtained a dichotomy result when each predicate is symmetric. In this work, we study Boolean PCSPs that contain the *simplest non-symmetric predicate*, $x \to y$. We call such Boolean PCSPs *Ordered* as we can also view the implication constraint as an ordering requirement $x \leq y$. We show that Ordered Boolean PCSPs exhibit computational dichotomy, assuming the recently introduced rich 2-to-1 conjecture (which is the perfect completeness analog of the Unique Games Conjecture) of Braverman, Khot, and Minzer [BKM21].

d-to-1 hardness of Coloring 3-Colorable graphs with $O(1)$ Colors. Approximate graph coloring is a canonical PCSP, and it is a major open problem to show NP-Hardness of coloring 3-colorable graphs with $O(1)$ colors. In this work, we prove that the d-to-1 conjecture for any fixed d implies the hardness of coloring a 3-colorable graph with C colors for arbitrarily large integers C. Here, the d-to-1 conjecture of Khot [Kho02b] asserts that it is NP-hard to satisfy an ϵ fraction of constraints of a satisfiable d-to-1 Label Cover instance, for arbitrarily small $\epsilon > 0$. Earlier, the hardness of $O(1)$-coloring a 4-colorable graphs is known [DMR09] under the 2-to-1 conjecture, which is the strongest in the family of d-to-1 conjectures, and the hardness for 3-colorable graphs is known under a certain "fish-shaped" variant of the 2-to-1 conjecture.

Rainbow Coloring Hardness via low sensitivity polymorphisms. Rainbow coloring of hypergraphs is another PCSP that, for most parameters, is harder than graph coloring, and thus, is easier to show unconditional NP-Hardness. Furthermore, it serves as a good testbed to analyze the polymorphisms of a PCSP that is similar to graph coloring. A k-uniform hypergraph is said to be r-rainbow colorable if there is an r-coloring of its vertices such that every hyperedge intersects all r color classes. Given as input such a hypergraph, finding a r-rainbow coloring of it is NP-hard for all $k \geq 3$ and $r \geq 2$. Therefore, one settles for finding a rainbow coloring with fewer colors (which is an easier task). When $r = k$ (the maximum possible value), i.e., the hypergraph is k-partite, one can efficiently 2-rainbow color the hypergraph, i.e., 2-color its vertices so that there are no monochromatic edges. In this work, we consider the next smaller value of $r = k - 1$, and prove that in this case, it is NP-hard to rainbow color the hypergraph with $q := \lceil \frac{k-2}{2} \rceil$ colors. In particular, for $k \leq 6$, it is NP-hard to 2-color $(k-1)$-rainbow colorable k-uniform hypergraphs.

Vector Bin Covering. We use the analysis of polymorphisms of rainbow coloring PCSP to prove the hardness of Vector Bin Covering. Vector Bin Covering is a multidimensional generalization of Bin Covering. In the d-dimensional Vector Bin Covering instance, the input is a set of n vectors in $[0, 1]^d$. The objective is to partition these into the maximum number of parts such that in each part, the sum of vectors is at least 1 in every coordinate. This problem is introduced by Alon *et al.* [Alo+98] who gave a $O(\log d)$ factor approximation algorithm. On the hardness front, Ray [Ray21] showed that the 2-dimensional Vector Bin Covering problem is hard to approximate within a factor of $\frac{998}{997}$. We show $\Omega\left(\frac{\log d}{\log \log d}\right)$ hardness for the problem, almost matching the $O(\log d)$ factor algorithm [Alo+98].

Robust algorithms for Promise CSPs. In this work, we study robust algorithms for PCSPs i.e., algorithms that output a solution satisfying $1 - f(\epsilon)$ fraction of the constraints when the instance is guaranteed to have a solution satisfying $1 - \epsilon$ fraction of constraints, where $f(\epsilon)$ goes to 0 as ϵ goes to 0. We show that if a Boolean folded PCSP contains Majority or Alternating-Threshold[1] family of polymorphisms, then it admits a robust algorithm. We also show Unique Games based hardness of obtaining robust algorithms for a broad class of PCSPs by showing integrality gaps for basic SDP [Rag08].

Revisiting Alphabet reduction. Along with Gap amplification, Alphabet reduction is a key step in Dinur's celebrated proof [Din07] of the PCP Theorem. The alphabet reduction used in [Din07] is proved via Assignment Testers. In this chapter, we give a simplified proof of alphabet reduction using a direct test, inspired by reductions between binary PCSPs.

In the rest of the three sections, we give an overview of the multidimensional packing problems, generalized Tuza's conjecture, and scheduling with non-uniform communication delay.

1.2 Multidimensional Packing and Scheduling

Vector Bin Packing is a multidimensional generalization of Bin Packing where the input is a set of n vectors in $[0, 1]^d$ and the goal is to partition the vectors into the minimum number of parts such that in each part, the sum of vectors is at most 1 in every coordinate. Already when $d = 2$, the problem is APX-hard [Woe97; Ray21]. On the algorithmic front, the PTAS for Bin Packing [VL81] easily implies a $d + \epsilon$ approximation for Vector Bin Packing. When d is part of the input, this is almost tight: there is a lower bound of $d^{1-\epsilon}$ shown by [CK04; Chr+17]. When d is

[1]Alternate-Threshold (AT) is a signed variant of the Majority function. For an odd integer L, $\mathrm{AT}_L(x_1, x_2, \ldots, x_L) = 1$, if $x_1 - x_2 + x_3 - \ldots + x_L > 0$, and 0 otherwise. If a PCSP Γ contains AT polymorphisms of all odd arities (for example, (1-in-3-SAT, NAE-3-SAT)), then it can be solved in polynomial time [BG21b].

a fixed constant[2], much better algorithms are known [CK04; BCS09; BEK16] that get $\ln d + O(1)$ approximation guarantee. However, the best hardness factor (for arbitrary constant d) is still the APX-hardness result of the 2-dimensional problem due to Woeginger from 1997. Closing this gap, either by obtaining a $O(1)$ factor algorithm or showing a hardness factor that is a function of d, has remained a challenging open problem.

We resolve this gap by proving a $\Omega(\log d)$ asymptotic hardness of approximation when d is a large constant, matching the $\ln d + O(1)$ approximation algorithms [CK04; BCS09; BEK16], up to constants. We obtain our hardness result via a reduction from the set cover problem on simple bounded set families – where the underlying set family is simple, each set has a bounded size, and each element appears in a bounded number of sets. We use this hardness to obtain the above theorem using a notion of embedding of set systems that we call *packing dimension*.

Vector Scheduling. Vector Scheduling is another well-studied problem for which the hardness of approximate graph coloring can be used to obtain almost optimal inapproximability results. In the d-dimensional Vector Scheduling problem, given a set of n vector jobs in $[0,1]^d$, and m identical machines, the objective is to assign the jobs to machines to minimize the maximum ℓ_∞ norm of the load on the machines. Chekuri and Khanna [CK04] introduced the problem as a natural generalization of Multiprocessor Scheduling and obtained a PTAS for the problem when d is a fixed constant. When d is part of the input, they obtained a $O(\log^2 d)$ factor approximation algorithm. They also showed that it is NP-hard to obtain a C factor approximation algorithm for the problem, for any constant C. Meyerson, Roytman, and Tagiku [MRT13] gave an improved $O(\log d)$ factor algorithm while the current best factor is $O\left(\frac{\log d}{\log\log d}\right)$ due to Harris and Srinivasan [HS19] and Im, Kell, Kulkarni, and Panigrahi [Im+19]. We show that these are almost tight by proving that the problem has no $\Omega((\log d)^{1-\epsilon})$ factor approximation algorithms assuming NP does not have quasipolynomial time algorithms. We also relate the problem to balanced hypergraph coloring PCSP and use this connection to show (albeit weaker) NP-Hardness result.

1.3 Approximate Hypergraph Vertex Cover and generalized Tuza's conjecture

A famous conjecture of Tuza [Tuz81; Tuz90] states that the minimum number of edges needed to cover all the triangles in a graph is at most twice the maximum number of edge-disjoint triangles. This conjecture was couched in a broader setting by Aharoni and Zerbib [AZ20] who proposed a hypergraph version of this conjecture and also studied its implied fractional versions. We establish the fractional version of the Aharoni-Zerbib conjecture up to lower-order terms. Specifically, we give a factor $t/2 + O(\sqrt{t \log t})$ approximation based on LP rounding for an algorithmic version of the hypergraph Turán problem (AHTP). The objective in AHTP is to pick the smallest collection of $(t-1)$-sized subsets of vertices of an input t-uniform hypergraph such that every hyperedge contains one of these subsets.

[2]The algorithms are now allowed to run in time $n^{f(d)}$, for some function f.

The algorithmic questions arising in the above study can be phrased as instances of vertex cover on *simple* hypergraphs, whose hyperedges can pairwise share at most one vertex. We prove that the trivial factor t approximation for vertex cover is hard to improve for simple t-uniform hypergraphs. However, for set cover on simple n-vertex hypergraphs, the greedy algorithm achieves a factor $(\ln n)/2$, better than the optimal $\ln n$ factor for general hypergraphs.

1.4 Scheduling with non-uniform communication delays

We study the problem of scheduling jobs with precedence and non-uniform communication delay constraints on identical machines to minimize the makespan objective function. This classic model was first introduced by Rayward-Smith [Ray87] and Papadimitriou and Yannakakis [PY90]. In this problem, we are given a set J of n jobs, where each job j has a processing length $p_j \in \mathbb{Z}_+$. The jobs need to be scheduled on m identical machines. The jobs have precedence and communication delay constraints, which are given by a partial order $\prec$. A constraint $j \prec j'$ encodes that job j' can only start after job j is completed. Moreover, if $j \prec j'$ and j, j' are scheduled on different machines, then j' can only start executing at least $c_{jj'}$ time units after j had finished. On the other hand, if j and j' are scheduled on the same machine, then j' can start executing immediately after j finishes. The goal is to schedule jobs *non-preemptively* to minimize the makespan objective function, which is defined as the completion time of the last job. In a non-preemptive schedule, each job j needs to be assigned to a single machine i and executed during a contiguous time interval of length p_j. In the classical scheduling notation, the problem is denoted by $\mathsf{P} \mid \mathsf{prec}, c_{jk} \mid C_{\mathsf{max}}$.

The problem has received renewed interest lately in the applied community due to its relevance in data center scheduling problems and large scale training of ML models. [Cho+11; Guo+12; GCL18; Tar+20]. From a theory perspective, very little was known other than the NP-Hardness of the problem. On the algorithmic front, recently, Maiti et al. [Mai+20] and Davies et al. [Dav+20; Dav+21] designed polylogarithmic approximation algorithms for the special case when all the communication delays are *equal*. On the hardness front, it remained an open problem whether the general problem containing arbitrary communication delays (referred to as non-uniform communication delay scheduling problem) has a constant factor approximation algorithm [SW99b; Ban17].

We answer this question in the negative by showing that for every $\epsilon > 0$, the the non-uniform communication delay problem ($\mathsf{P} \mid \mathsf{prec}, c_{jk} \mid C_{\mathsf{max}}$) does not admit a polynomial-time $(\log n)^{1-\epsilon}$-approximation algorithm assuming $\mathit{NP} \not\subseteq \mathit{ZTIME}\left(n^{(\log n)^{O(1)}}\right)$. Our proof of the result follows via a scheduling problem that we call Unique Machine Precedence constraints Scheduling (UMPS), which we believe is a fundamental problem on its own.

1.5 Chapter Credits

Chapter 4, Chapter 5 and Chapter 9 are based on joint works [GS20a], [GS20b] and [GS22] respectively, with Venkatesan Guruswami. Chapter 3 and Chapter 6 are based on [BGS21] and an unpublished work respectively, with Joshua Brakensiek and Venkatesan Guruswami. Chapter 7 is based on [GOS20] a joint work with Venkatesan Guruswami and Jakub Opršal. Chapter 8 is based on [San21]. Chapter 10 is based on [Dav+22] a joint work with Sami Davies, Janardhan Kulkarni, Thomas Rothvoss, Jakub Tarnawski and Yihao Zhang.

1.6 Organization

The thesis is divided into two parts.

In the first part, we study Promise CSPs. We start with some formal definitions in Chapter 2. Then, we prove the conditional dichotomy of Boolean Ordered PCSPs in Chapter 3. We study the approximate graph coloring and rainbow coloring problems in Chapter 4 and Chapter 5 respectively. The robust algorithms for PCSPs are studied in Chapter 6. Finally, we study the alphabet reduction in Chapter 7.

In the second part, we study multidimensional packing problems in Chapter 8, generalized Tuza's conjecture in Chapter 9, and scheduling with non-uniform communication delay problem in Chapter 10. We conclude with some open directions in Chapter 11.

Part I

Promise Constraint Satisfaction Problems

Chapter 2

Promise Constraint Satisfaction Problems: Introduction

In this chapter, we introduce Promise Constraint Satisfaction Problems(PCSPs) formally.

2.1 PCSPs and Polymorphisms.

Constraint satisfaction problems (CSP) have played a very influential role in the theory of computation, providing an excellent testbed for the development of both algorithmic and hardness techniques, which then extend to more general settings. A CSP over domain D is specified by a finite collection of predicates over D. Given an input containing n variables with constraints on the variables using these predicates, the objective is to identify if we can assign values from D to the variables that satisfies all the constraints.

Definition 1. *(CSP) Given a k-ary relation $A : D^k \to \{0,1\}$ over a domain D, the Constraint Satisfaction Problem(CSP) associated with the predicate A takes a set of variables $V = \{v_1, v_2, \dots, v_n\}$ as input which are to be assigned values from D. There are m constraints $(e_1, e_2, \dots, e_m)$ each consisting of $e_i = ((e_i)_1, (e_i)_2, \dots, (e_i)_k) \subseteq V^k$ that indicate that the corresponding assignment should belong to A. The objective is to identify if there is an assignment $V \to D$ that satisfies all the constraints.*

In general, we can have multiple relations $A_1, A_2, \dots, A_l$, and different constraints can use different relations. We denote such a CSP by $CSP(A_1, A_2, \dots, A_l)$.

We give some examples of CSPs.

Example 2. *Examples of CSPs include*

1. 3-*SAT: In this problem, the objective is to assign True or False to variables to satisfy a Boolean formula that is a conjunction of clauses each of which is a disjunction of three literals. In the above notation, we have two predicates, the 3-ary predicate $x \vee y \vee z$ together with allowing negation of variables i.e., the predicate $y = \overline{x}$.*
2. 2-*SAT: This corresponds to the Boolean formula satisfiability problem when each clause contains two literals.*

3. 3-coloring. *In this problem, given a graph G, the objective is to assign 3 colors to the vertices of G such that every pair of adjacent vertices are assigned different colors. In the above notation, we have a single predicate $A = \{(x, y) : x, y \in [3], x \neq y\}$.*
4. 1-*in*-3-*SAT. In this problem, we are given a set of constraints involving three variables, and the objective is to assign True or False to the variables such that in constraint, exactly one variable is assigned True. In the above notation, we have a single predicate $A = \{(x, y, z) : x, y, z \in \{0, 1\}, x + y + z = 1\}$.*
5. *NAE-3-SAT. In this problem, we have a set of constraints involving three variables over the Boolean domain, and the objective is to assign True or False such that in each constraint, both True and False appear. The predicate is $A = \{0, 1\}^n \setminus \{(0, 0, 0), (1, 1, 1)\}$.*

The key computational challenge in the study of CSPs is to characterize the computational complexity of them i.e., identify which CSPs can be solved in polynomial time, which CSPs are NP-Hard, and whether every CSP is in P or is NP-Hard. In the above set of examples, 2-SAT can be solved in polynomial time while the rest of the CSPs are NP-Hard. The formal study of CSPs was initiated by Schaefer [Sch78] in 1978 when he proved that every Boolean CSP is either in P or is NP-Complete. Feder and Vardi [FV98] conjectured that the same should hold over arbitrary domains as well. After a long line of works, the Feder-Vardi conjecture was resolved in the affirmative by Bulatov [Bul17] and Zhuk [Zhu20] independently.

In this thesis, we study Promise Constraint Satisfaction Problems (PCSPs) that vastly generalize CSPs. In the PCSPs, each predicate has a weak and a strong form–given an instance of PCSP containing n variables with the constraints, the goal is to distinguish between the case that the stronger form can be satisfied vs. even the weaker one cannot be satisfied. We formally define Promise Constraint Satisfaction Problems(PCSPs).

Definition 3. *(PCSP) In a Promise Constraint Satisfaction Problem $PCSP(\Gamma)$ over a pair of domains D_1, D_2, we have a set of pairs of relations $\Gamma = \{(A_1, B_1), (A_2, B_2), \ldots, (A_l, B_l)\}$ such that for every $i \in [l]$, A_i is a subset of $D_1^{k_i}$ and B_i is a subset of $D_2^{k_i}$. Furthermore, there is a homomorphism $h : D_1 \to D_2$ such that for all $i \in [l]$ and $x \in D_1^{k_i}$, $x \in A_i$ implies $h(x) \in B_i$. Given a $CSP(A_1, A_2, \ldots, A_l)$ instance, the objective is to distinguish between the two cases:*

1. *There is an assignment to the variables from D_1 that satisfies every constraint when viewed as $CSP(A_1, A_2, \ldots, A_l)$.*
2. *There is no assignment to the variables from D_2 that satisfies every constraint when viewed as $CSP(B_1, B_2, \ldots, B_l)$.*

We give some examples of PCSPs.

Example 4. *Examples of PCSPs:*

1. (c, s)*-approximate Graph Coloring. A classical example of PCSP is the approximate graph coloring, where given a graph G, the goal is to distinguish between the cases that G can be colored with c colors vs. it cannot be colored with s colors for some $c \leq s$. In the above language, we have predicates A, B where $A = \{(x, y) : x, y \in [c], x \neq y\}$, and $B = \{(x, y) : x, y \in [s], x \neq y\}$.*
2. *(1-in-3-SAT, NAE-3-SAT). Given a 1-in-3-SAT instance that is promised to be satisfiable, the objective is to assign $0, 1$ values to the variables such that each constraint is satisfied as*

an NAE-3-SAT instance, i.e., both 0 and 1 occur in every constraint.

3. $(2+\epsilon)-SAT$*: Given a CNF formula where each clause has w literals, with the promise that there is an assignment satisfying $\lceil \frac{w}{2} \rceil - 1$ literals in each clause, the objective is to find a satisfying assignment to the formula.*
4. *Approximate rainbow coloring of hypergraphs: A k uniform hypergraph is said to be r-rainbow colorable if there exists a coloring to the vertices such that in each edge, all the colors appear. Given a k-uniform hypergraph that is promised to be $(k-1)$-rainbow colorable, the objective is to 2-color[1] the hypergraph.*

The study of PCSPs was formally initiated by Austrin, Guruswami, and Håstad [AGH17]. and since then, there has been a lot of recent interest in PCSPs, including the development of a systematic theory in [BG21a; Bar+21] and leading to breakthroughs in approximate graph coloring [Bar+21; KO19; WŽ20]. Similar to CSPs, the key question in the study of PCSPs is to understand which PCSPs can be solved in polynomial time, which ones are NP-Hard, and if there is a PCSP dichotomy result. Consider the above example of (1-in-3-SAT, NAE-3-SAT). While the individual CSPs, namely 1-in-3-SAT and NAE-3-SAT are both NP-hard, the above PCSP indeed can be solved in polynomial time [BG21a]. On the other hand, (c, s)-approximate graph coloring is much less understood: it's only in a recent breakthrough result [Bar+21] that $(3, 5)$-approximate graph coloring is shown to be NP-Hard.

As is the case with CSPs, the key tool underlying these results is the universal algebraic framework where polymorphisms of the PCSP are studied. Polymorphisms capture the closure properties of the satisfying solutions to the PCSP. More formally, we can define polymorphisms of a PCSP as follows.

Definition 5. *(Polymorphisms) For $PCSP(\Gamma)$ with $\Gamma = \{((A_1, B_1), (A_2, B_2), \ldots, (A_l, B_l))\}$ where for every $i \in [l]$, $A_i : [q_1]^{k_i} \to \{0,1\}$, $B_i : [q_2]^{k_i} \to \{0,1\}$, a polymorphism of arity n is a function $f : [q_1]^n \to [q_2]$ that satisfies the below property for all $i \in [l]$. For all $(\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_{k_i})$ such that for all $j \in [n]$, $((\boldsymbol{v}_1)_j, (\boldsymbol{v}_2)_j, \ldots, (\boldsymbol{v}_{k_i})_j) \in A_i$, we have*

$$(f(\boldsymbol{v}_1), f(\boldsymbol{v}_2), \ldots, f(\boldsymbol{v}_{k_i})) \in B_i$$

We use $\mathrm{Pol}(\Gamma)$ *to denote the family of all the polymorphisms of $PCSP(\Gamma)$.*

We refer the reader to [Bar+21] for an extensive introduction to PCSPs and polymorphisms.

2.2 Label Cover

We now formally define the Label Cover problem that serves as a starting point for most of our hardness results.

Definition 6. *(Label Cover) In the Label Cover instance, we are given a tuple $G = ((V, E), R, \Psi)$ where*

1. *(V, E) is a graph on vertex set V with edge set E.*
2. *Each vertex in V has to be assigned a label from the set $\Sigma = [R] = \{1, 2, \ldots, R\}$.*

[1]Rainbow 2-coloring is the same as standard 2-coloring of hypergraphs.

3. *For every edge $e = (u, v) \in E$, there is an associated relation $\Psi_e \subseteq \Sigma \times \Sigma$. This corresponds to a constraint between u and v.*

A labeling $\sigma : V \to \Sigma$ satisfies a constraint associated with the edge $e = (u, v)$ if and only if $(\sigma(u), \sigma(v)) \in \Psi_e$. Given such an instance, the goal is to distinguish if there is a labeling that can satisfy all the constraints or if no labeling can satisfy a significant fraction of constraints.

The PCP theorem [AS98; Aro+98] together with Raz's parallel repetition theorem [Raz98] implies that for every constant $\epsilon > 0$, it is NP-hard to distinguish between the case that a given Label Cover instance has a labeling that satisfies all the constraints vs. no labeling can satisfy more than ϵ fraction of the constraints. This hardness result for Label Cover has been instrumental in showing numerous strong, and sometimes optimal, inapproximability results for various computational problems.

Chapter 3

Conditional dichotomy of Boolean Ordered PCSPs

3.1 Introduction

Towards establishing a potential Boolean PCSP dichotomy, progress has been made by Ficak, Kozik, Olsák and Stankiewicz [Fic+19], who obtained a dichotomy result when each predicate is symmetric. In this chapter, we study Boolean PCSPs that contain the *simplest non-symmetric predicate*, $x \to y$. We call such Boolean PCSPs *Ordered* as we can also view the implication constraint as an ordering requirement $x \le y$.

Ordered Boolean PCSPs have come under recent study. The work of Petr [Pet20] (inspired by work of Barto [Bar18b; Bar18a]) considered a special class of Ordered Boolean PCSPs which have an additional predicate $x \neq y$ (this corresponds to allowing negations in the constraints) as well as the requirement that the majority on three bits is *not* a polymorphism. In this setting Petr was able to show that such Ordered Boolean PCSPs are NP-hard. However, the approach considered does not seem immediately extendable to analyzing general Ordered Boolean PCSPs [Bar18a].

The main motivation for studying these PCSPs comes from the fact that adding the additional $x \le y$ predicate is equivalent to restricting the polymorphisms of the PCSPs to be *monotone functions*. Monotonicity is an influential theme in the study of Boolean functions and complexity theory, and understanding the structure of polymorphisms in the monotone case is an important (and certainly necessary) subcase towards a general characterization of polymorphisms vs. tractability for arbitrary Boolean PCSPs. For the special case of Boolean Ordered PCSPs which include negation constraints, it was conjectured in [Bar18a] that polynomial time tractability is characterized by the existence of majority polymorphisms of arbitrarily large arity.

Our main result is that Boolean Ordered PCSPs exhibit a dichotomy, under the recently introduced *Rich* 2*-to-*1 *Conjecture* of Braverman, Khot, and Minzer [BKM21].

Theorem 7. *Assuming the Rich* 2*-to-*1 *Conjecture, every Ordered Boolean PCSP is either in P or is NP-Complete. Furthermore, an Ordered PCSP* Γ *is in P if and only if for every* $\epsilon > 0$*, there are polymorphisms of* Γ *with every coordinate having Shapley value at most* ϵ*. Equivalently,* Γ *is in P if and only if it has threshold polymorphisms of arbitrarily large arity.*

As a concrete example, recall the PCSP (1-in-3-SAT, NAE-3-SAT) defined in Chapter 2. As it has threshold polymorphisms of arbitrarily large arity, it remains polynomial time solvable even after adding the predicate $x \to y$. However, if we also add another two-variable predicate $x \neq y$, the PCSP no longer has threshold polymorphisms, and by our above result, it becomes NP-Complete.

We obtain the conditional dichotomy result by analyzing the polymorphisms of the Ordered PCSPs. The key idea in the algebraic approach to PCSPs is that the PCSP is tractable if the polymorphisms are close to symmetric, and the PCSP is hard if all the polymorphisms have a small number of "important" coordinates. More concretely, on the algorithmic front, it has been proved that symmetric polymorphisms of arbitrarily large arities lead to polynomial time algorithms for PCSPs [Bra+20]. On the hardness side, if all the polymorphisms depend on a bounded number of coordinates, then the underlying PCSP is NP-hard [AGH17]. This has been extended to various other notions, including combinatorial ones such as C-fixing [BG16], and topological ones such as having a bounded number of coordinates with non-zero winding number [KO19]. In this work, we study the monotone polymorphisms using analytical techniques.

In particular, we use Shapley value to analyze the monotone polymorphisms. For a monotone function $f : \{0,1\}^n \to \{0,1\}$, Shapley value of a coordinate i is the probability that on a random path from $\{0, 0, \ldots, 0\}$ to $\{1, 1, \ldots, 1\}$, the function value turns from 0 to 1 when we switch the ith coordinate to 1. Initially studied to understand the power of an individual in voting systems [SS54], Shapley value has now found applications in various settings, especially in game theory [Mic+13; NN11]. In our setting, there are two advantages of using Shapley value to study the polymorphisms. First, it is a relative measure of the importance of a coordinate, as opposed to other notions of Influence which are absolute. This helps in bounding the number of coordinates with Shapley value above a certain threshold. Second, it is a versatile measure with combinatorial and analytical interpretations [DDS17] which helps in proving that Shapley value stays consistent under function minors[1], a key property necessary in both the algorithm and the hardness.

Algorithm Overview. We obtain our algorithmic result by using the Basic Linear Programming with Affine relaxation (BLP+Affine relaxation), combined with a structural result regarding the monotone functions with bounded Shapley value. As mentioned earlier, PCSPs with symmetric polymorphisms of arbitrarily large arities can be solved in polynomial time using the BLP+Affine relaxation algorithm [Bra+20]. Our main structural result is that Boolean functions with bounded Shapley value have arbitrarily large threshold functions as minors. Since the set of polymorphisms of a PCSP are closed under taking minors, this proves that the underlying PCSP Γ has arbitrarily large threshold functions as polymorphisms, which then implies that Γ is in P. The key tool underlying our structural result is a result of Kalai [Kal04] that states that under certain conditions, monotone Boolean functions with arbitrarily small Shapley value have a sharp threshold.

Hardness Overview. We obtain our hardness result assuming the Rich 2-to-1 Conjecture. Braverman, Khot, and Minzer [BKM21] introduced the conjecture as a perfect completeness surrogate of the well known Unique Games Conjecture [Kho02b]. They also proved that the conjecture is

[1]A minor(formally defined in Section 3.2) of a function $f : \{0,1\}^m \to \{0,1\}$ is a function $g : \{0,1\}^n \to \{0,1\}$ of smaller arity $n \leq m$ obtained from f by identifying sets of variables together.

equivalent to Unique Games Conjecture when we relax the perfect completeness requirement. The reduction from the Rich 2-to-1 Conjecture to PCSPs follows using the standard Label Cover-Long Code paradigm. The key ingredient in this reduction is a decoding of the Long Codes to a bounded number of coordinates that is consistent under function minors. We decode each Long Code function to the coordinates with $\Omega(1)$ Shapley value—as the sum of Shapley values of all the coordinates of any monotone function is equal to 1, there is a bounded number of such coordinates. We argue about the consistency of this decoding using a structural result that states that under a uniformly random minor, Shapley value is roughly preserved.

On the necessity of "richness" in 2-to-1 Conjecture. A natural question is whether our hardness result can be obtained using a weaker assumption such as the 2-to-1 conjecture (whose imperfect completeness version was recently established [KMS17; Din+18a; Din+18b; KMS18]). We shed some light on this question by showing that there are monotone Boolean functions $f : \{0,1\}^{2n} \to \{0,1\}$ and $g : \{0,1\}^n \to \{0,1\}$ such that g is a minor of f with respect to the 2-to-1 function π, both the functions f and g have exactly one coordinate i_1, i_2 respectively, with $\Omega(1)$ Shapley value, and yet $\pi(i_1) \neq i_2$. Such an adversarial example is interesting from two angles: first, it shows that even using the 2-to-1 conjecture, the Shapley value based decoding is not consistent. Second, it gives an example of agents pairing up maliciously to completely alter the Shapley value. The underlying phenomenon is that the rich 2-to-1 games have "subcode-covering" property, which is absent in the standard 2-to-1 games, helping in preserving the consistency of any biased influence measure such as the Shapley value.

Organization. In Section 3.2, we formally define PCSPs, polymorphisms, and Shapley value. We present the algorithmic and hardness parts of our dichotomy result in Section 3.3 and Section 3.4 respectively. We present the adversarial example of a 2-to-1 minor that alters the Shapley value in Section 3.5.

3.2 Preliminaries

Notations. We use $[n]$ to denote the set $\{1, 2, \ldots, n\}$. For a k-ary relation $A \subseteq [q]^k$, we abuse the notation and use A both as a subset of $[q]^k$, and also as a predicate $A : [q]^k \to \{0,1\}$. For a vector $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in \{0,1\}^n$, we use $\mathsf{hw}(\mathbf{x})$ to denote $\sum_{i=1}^n x_i$. For two vectors $\mathbf{x}, \mathbf{y} \in \{0,1\}^n$, we say that $\mathbf{x} \leq \mathbf{y}$ if $x_i \leq y_i$ for all $i \in [n]$. A Boolean function $f : \{0,1\}^n \to \{0,1\}$ is called monotone if $f(\mathbf{x}) \leq f(\mathbf{y})$ for all $\mathbf{x} \leq \mathbf{y}$.

Boolean Ordered PCSPs and Minors of functions. We formally define Boolean Ordered PCSPs.

Definition 8. *(Boolean Ordered PCSP) A PCSP $PCSP(\Gamma)$ over a pair of domains D_1, D_2 with the set of pairs of relations $\Gamma = \{(A_1, B_1), (A_2, B_2), \ldots, (A_l, B_l)\}$ is said to be Boolean Ordered if the following hold.*

1. *The domains are both Boolean i.e., $D_1 = D_2 = \{0,1\}$.*
2. *There exists $i \in [l]$ such that $A_i = B_i = \{(0,0), (0,1), (1,1)\}$.*

A crucial property satisfied by the polymorphisms of a PCSP Γ, $\mathrm{Pol}(\Gamma)$ is that the family of functions is closed under taking minors. We first define the minor of a function formally.

Definition 9. *(Minor of a function) For a Boolean function $f : [q]^n \to [q']$, the function $g : [q]^m \to [q']$ is said to be a minor of f with respect to the function $\pi : [n] \to [m]$ if*

$$g(x_1, x_2, \ldots, x_m) = f(x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(n)})\, \forall x_1, x_2, \ldots, x_m \in [q]$$

We say that a function g is a minor of f if there exists some π such that g is a minor of f with respect to π.

We are often interested in 2-to-1 minors. A function g is said to be a 2-to-1 minor of f if there exists a 2-to-1 function π such that g is a minor of f with respect to π, where 2-to-1 function is defined below.

Definition 10. *(2-to-1 function) A function $\pi : [2n] \to [n]$ is said to be a 2-to-1 function if*

$$|\pi^{-1}(i)| = 2\ \forall i \in [n]$$

We use $\mathcal{F}_{2\to 1}(n)$ to denote the set of all the 2-to-1 functions from $[2n]$ to $[n]$.

By the definition of the polymorphisms, we can infer that if $f \in \mathrm{Pol}(\Gamma)$ for a PCSP Γ, then for all functions g such that g is a minor of f, we have $g \in \mathrm{Pol}(\Gamma)$. Such a family of functions that is closed under taking minors is called as a *minion*. We often refer to the family of polymorphisms of a PCSP as the polymorphism minion.

Shapley value. Let $f : \{0,1\}^n \to \{0,1\}$ be a monotone Boolean function. We can view the monotone Boolean function f as a voting scheme between two parties, and n agents: the winner of the voting scheme when the ith agent votes for $\mathbf{x}_i \in \{0,1\}$ is $f(\mathbf{x})$. The relative power of an agent in a voting scheme is typically measured using the Shapley-Shubix Index, also known as Shapley Value.

Informally speaking, the Shapley Value of a coordinate i is the probability that the ith agent is the altering vote when we start with all zeroes and flip the votes in a uniformly random order. More formally,

Definition 11. *(Shapley value) Let $f : \{0,1\}^n \to \{0,1\}$ be a monotone Boolean function. Let $\sigma \in S_n$ be a uniformly random permutation of $[n]$. For an integer $j \in [n]$, let P_j denote the set of first j elements of σ i.e., $P_j := \{\sigma(1), \sigma(2), \ldots, \sigma(j)\}$. The Shapley value $\Phi_f(i)$ of the coordinate $i \in [n]$ is defined as*

$$\Phi_f(i) := \mathbf{Pr}_\sigma \{\exists j \in [n] : \sigma(j) = i, f(P_{j-1}) = 0, f(P_j) = 1\}$$

We also give an alternate definition of Shapley value using the notion of boundary of a coordinate. For a monotone Boolean function $f : \{0,1\}^n \to \{0,1\}$ and coordinate $i \in [n]$, let $\mathcal{B}_f(i)$ denote the boundary of the coordinate i i.e.

$$\mathcal{B}_f(i) := \{S \subseteq [n] \setminus \{i\} : f(\{i\} \cup S) = 1, f(S) = 0\}$$

By the monotonicity of f, we can infer that $\mathcal{B}_f(i)$ satisfies the following sandwich property that will be useful later.

Proposition 12. *Let $f : \{0,1\}^n \to \{0,1\}$ be a monotone Boolean function and let $i \in [n]$. Then, for every pair of sets $S_1, S_2 \in \mathcal{B}_f(i)$ with $S_1 \subseteq S_2$, we have $S \in \mathcal{B}_f(i)$ for all S such that $S_1 \subseteq S \subseteq S_2$.*

Proof. By the monotonicity of f, we have $f(S \cup \{i\}) \geq f(S_1 \cup \{i\}) = 1$, and thus, $f(S \cup \{i\}) = 1$. Similarly, we have $f(S) \leq f(S_2) = 0$, and thus, $f(S) = 0$. $\square$

For an index $j \in \{0, 1, \ldots, n-1\}$, let $\mu_f(j)^{(i)}$ denote the fraction of subsets of $[n]$ of size j that are in $\mathcal{B}_f(i)$ i.e.

$$\mu_f(j)^{(i)} := \left| \mathcal{B}_f(i) \cap \binom{[n]}{j} \right| \Big/ \binom{n}{j}.$$

We can rewrite the definition of Shapley value of the ith coordinate as the following [Web77]:

$$\Phi_f(i) = \frac{\sum_{j=0}^{n-1} \mu_f(j)^{(i)}}{n} \,. \tag{3.1}$$

3.3 Algorithm when Shapley values are small

In this section, we show that monotone Boolean functions where each coordinate has bounded Shapley value has arbitrarily large threshold functions as minors, thereby proving the algorithmic part of our dichotomy result.

Let L be a positive integer and $0 \leq \tau \leq L$ be a non-negative integer. We let $\mathsf{THR}_{L,\tau} : \{0,1\}^L \to \{0,1\}$ be the threshold function on L variables with threshold τ. More formally,

$$\mathsf{THR}_{L,\tau}(\mathbf{x}) := \begin{cases} 1 \text{ if } \mathsf{hw}(\mathbf{x}) \geq \tau \\ 0 \text{ otherwise.} \end{cases}$$

For a monotone Boolean function $f : \{0,1\}^n \to \{0,1\}$ and real number $p \in [0,1]$, let $P_p(f)$ denote the expected value of $f(x)$ where each element $x_i, i \in [n]$ is independently set to be 1 with probability p and 0 with probability $1-p$. For every monotone function f, the function $P_p(f)$ is a strictly monotone continuous function in p on the interval $[0,1]$. The value $p_c = p_c(f)$ at which $P_{p_c}(f) = \frac{1}{2}$ is called the *critical probability* of f.

Using the Russo-Margulis Lemma [Rus82; Mar74] and Poincaré Inequality, we can show the following lemma that we need later.

Lemma 13 (Exercise 8.29(e) in [ODo14])**.** *Let f be a non-constant monotone Boolean function with critical probability $p_c \leq \frac{1}{2}$. Let $p_1 := \frac{1}{(2\nu)^2} p_c$ for $\nu > 0$. If $p_1 \leq \frac{1}{2}$, then $P_{p_1}(f) \geq 1 - \nu$.*

We now define the threshold interval of f.

Definition 14. *For a monotone function f and $0 < \epsilon < \frac{1}{2}$, we define $T_\epsilon(f) := p_2 - p_1$, where p_2 and p_1 are such that $P_{p_1}(f) = \epsilon, P_{p_2}(f) = 1 - \epsilon$.*

Kalai [Kal04] proved the following result regarding monotone Boolean functions.

Theorem 15. *For every $a, \epsilon, \gamma > 0$, there exists $\delta := \delta(a, \epsilon, \gamma) > 0$ such that for every monotone Boolean function $f : \{0,1\}^n \to \{0,1\}$ with $\Phi_f(i) \leq \delta$ for all $i \in [n]$ and $a \leq p_c(f) \leq 1 - a$, then $T_\epsilon(f) \leq \gamma$.*

We will use this result to show that for every monotone function where each coordinate has bounded Shapley value has arbitrarily large threshold functions as minor.

Lemma 16. *For every $L \geq 2$, there exists a $\delta := \delta(L) > 0$ such that the following holds. For any monotone Boolean function $f : \{0,1\}^n \to \{0,1\}$ with*

$$\Phi_f(i) \leq \delta \ \forall i \in [n]$$

there exists a positive integer $L' \in \{L, L+1\}$ and a non-negative integer τ such that $\mathsf{THR}_{L',\tau}$ is a minor of f.

Proof. We obtain $\delta := \delta(L) > 0$ from Theorem 15 by setting $\epsilon = \frac{1}{2^{L+1}}, \gamma = a = \frac{1}{L^3}$. Our goal is to show that for this parameter δ, for every monotone Boolean function f with each coordinate having Shapley value at most δ, there exists $L' \in \{L, L+1\}$ and τ such that $\mathsf{THR}_{L',\tau}$ is a minor of f.

We assume that f is a non-constant function, else we have a trivial minor by setting $\tau = 0$ or $\tau = L'$. Let p_c be the critical probability of f.

Case 1: $p_c < a = \frac{1}{L^3}$.

Let $p_1 = L^2 p_c < \frac{1}{L}$. Using Lemma 13, we can conclude that $P_{p_1}(f) \geq 1 - \frac{1}{2L}$. As $P_p(f)$ is monotone, we get that $P_{\frac{1}{L}}(f) > 1 - \frac{1}{2L}$. We let $g : \{0,1\}^L \to \{0,1\}$ be a uniformly random minor of f i.e. we choose the function $\pi : [n] \to [L]$ by choosing each value $\pi(i)$ uniformly and independently at random from $[L]$, and we let g to be the minor of f with respect to π.

Note that for every $i \in [L]$, the distribution of $g(\{i\})$ over the random minor g is the same as sampling a random input to f where we set each bit to 1 with probability $\frac{1}{L}$. As $P_{\frac{1}{L}}(f) \geq 1 - \frac{1}{2L}$, we get that for each $i \in [L]$, $g(\{i\}) = 1$ with probability at least $1 - \frac{1}{2L}$. By union bound, with probability at least $\frac{1}{2}$, $g(\{i\}) = 1$ for all $i \in [L]$. As $f(0,0,\ldots,0) = 0$, $g(\phi) = 0$ as well. Thus, with probability at least $\frac{1}{2}$, $g = \mathsf{THR}_{L,1}$. Hence, $\mathsf{THR}_{L,1}$ is a minor of f.

Case 2: $p_c > 1 - a = 1 - \frac{1}{L^3}$.

Let $f^\dagger$ be the Boolean dual of f defined as $f^\dagger(x) = 1 - f(\overline{x})$. Note that $P_p(f^\dagger) = 1 - P_{1-p}(f)$ for all $p \in [0,1]$. Thus, $p_c(f^\dagger) = 1 - p_c < a$. Using the previous case, we can infer that $\mathsf{THR}_{L,1}$ is a minor of $f^\dagger$ with respect to a funtion $\pi : [n] \to [L]$. The same function π proves that $\mathsf{THR}^\dagger_{L,1} = \mathsf{THR}_{L,L}$ is a minor of f.

Case 3: $a \leq p_c \leq 1 - a$.

Using Theorem 15, we obtain p_1 such that $P_{p_1}(f) \leq \epsilon$, and $P_{p_1+\gamma} \geq 1 - \epsilon$, where $\epsilon = \frac{1}{2^{L+1}}, \gamma = \frac{1}{L^3}$. As $\gamma < \frac{1}{L(L+1)}$, there exists $L' \in \{L, L+1\}$ and $\tau \in [L']$ such that $p_1 + \gamma < \frac{\tau}{L'}$ and $p_1 > \frac{\tau-1}{L'}$. Thus, we get that $P_{\frac{\tau}{L'}}(f) > 1 - \epsilon$ and $P_{\frac{\tau-1}{L'}} < \epsilon$. Let $g : \{0,1\}^{L'} \to \{0,1\}$ be a uniformly random minor of f i.e. we choose $\pi : [n] \to [L']$ by setting each value uniformly and independently at random from $[L']$ and set g to be the minor of f with respect to π. For a vector $\mathbf{x} \in \{0,1\}^{L'}$ with $\mathsf{hw}(\mathbf{x}) = \tau$, with probability greater than $1 - \frac{1}{2^{L+1}}$, $g(\mathbf{x}) = 1$. Similarly, for $\mathbf{x} \in \{0,1\}^{L'}$ with $\mathsf{hw}(\mathbf{x}) = \tau - 1$, with probability greater than $1 - \frac{1}{2^{L+1}}$, $g(\mathbf{x}) = 0$. Thus, with non-zero probability, $g(\mathbf{x}) = 1$ for all $x \in \{0,1\}^{L'}$ with $\mathsf{hw}(\mathbf{x}) = \tau$ and $g(\mathbf{x}) = 0$ for all $\mathbf{x} \in \{0,1\}^{L'}$ with $\mathsf{hw}(\mathbf{x}) = \tau - 1$. In other words, with non-zero probability, g is equal to $\mathsf{THR}_{L',\tau}$. Thus, $\mathsf{THR}_{L',\tau}$ is a minor of f. $\square$

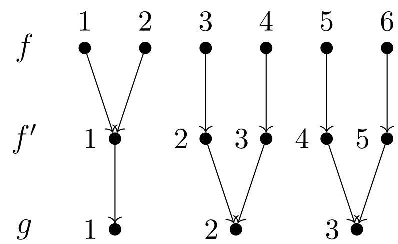

Figure 3.1: An illustration of the two step minor approach: Here $f : \{0,1\}^6 \to \{0,1\}$ is a Boolean function, $f' : \{0,1\}^5 \to \{0,1\}$ is a minor of f with respect to the function $\pi_1 : [6] \to [5]$ with $\pi_1(i) = \max(i-1, 1)$, and g is a minor of f' with respect to the function $\pi_2 : [5] \to [3]$ with $\pi_2(i) = \lceil \frac{i+1}{2} \rceil$.

Using the existence of arbitrarily large arity threshold minors, the algorithmic part of our Dichotomy result follows immediately.

Theorem 17. *Let Γ be a Promise CSP template. Suppose that for every $\epsilon > 0$, there exists a function $f \in \mathrm{Pol}(\Gamma), f : \{0,1\}^n \to \{0,1\}$ such that $\Phi_i(f) \leq \epsilon$ for all $i \in [n]$. Then,* $\mathrm{PCSP}(\Gamma) \in \mathsf{P}$.

Proof. Using Lemma 16, we can conclude that there are infinitely many positive integers L such that there exists $\tau \in \{0, 1, \ldots, L\}$ with $\mathsf{THR}_{L,\tau} \in \mathrm{Pol}(\Gamma)$. As the threshold functions are symmetric[2], $\mathrm{Pol}(\Gamma)$ has symmetric polymorphisms of infinitely many arities. Thus, using the BLP+Affine algorithm of [Bra+20], $\mathrm{PCSP}(\Gamma)$ can be solved in polynomial time. $\square$

We remark that the above result is inspired by a special case shown by Barto [Bar18b] that a Boolean Ordered PCSP is polytime tractable if it has cyclic polymorphisms of arbitrarily large arities.

3.4 Hardness Assuming Rich 2-to-1 Conjecture

In this section, we prove the hardness part of our dichotomy result. First, we prove that Shapley value is preserved under uniformly random 2-to-1 minors, and then we use this to show the hardness assuming the Rich 2-to-1 Conjecture.

3.4.1 Shapley value under random 2-to-1 minor

Let $f : \{0,1\}^{2n} \to \{0,1\}$ be a monotone Boolean function with $\Phi_f(1) \geq \lambda$ for some absolute constant $\lambda > 0$. Let $g : \{0,1\}^n \to \{0,1\}$ be a minor of f with respect to the uniformly random 2-to-1 function $\pi : [2n] \to [n]$. Our goal in this subsection is to show that $\mathbb{E}_\pi[\Phi_g(\pi(1))] \geq \gamma$ for some function $\gamma := \gamma(\lambda) > 0$. We prove this in two steps. (See Figure 3.1)

[2]A function $f : \{0,1\}^n \to \{0,1\}$ is said to be symmetric if it is unchanged by any permutation of the input variables.

1. First, we consider the minor of f, $f' : \{0,1\}^{2n-1} \to \{0,1\}$ obtained with respect to $\pi_1 : [2n] \to [2n-1]$ where $\pi_1(1) = \pi_1(2) = 1, \pi_1(i) = i-1 \, \forall i \in \{3,4,\ldots,2n\}$. We show that $\Phi_{f'}(1) \geq \frac{\lambda}{2}$.
2. Next, we consider a minor g of f' obtained with respect to the function $\pi_2 : [2n-1] \to [n]$ which has $\pi_2(1) = 1$ while the rest $2n-2$ values are chosen using a uniformly random partition of $[2n-2]$ into $n-1$ pairs. We show that $\mathbb{E}_{\pi_2}[\Phi_g(1)] \geq \gamma$ for some function $\gamma := \gamma(\lambda) > 0$.

Note that the process of first taking the f' minor and then obtaining g by partitioning $[2n-2]$ into $n-1$ uniformly random pairs is equivalent to taking a uniformly random 2-to-1 minor of f. Thus, the two steps together prove the required Shapley value property of the uniformly random 2-to-1 minor.

The first step is captured by the following lemma.

Lemma 18. *Let $f : \{0,1\}^{2n} \to \{0,1\}$ and $f' : \{0,1\}^{2n-1} \to \{0,1\}$ be monotone Boolean functions such that f' is a minor of f with respect to the function $\pi_1 : [2n] \to [2n-1]$ defined as $\pi_1(i) = \max(i-1, 1)$. If $\Phi_f(1) \geq \lambda$, then $\Phi_{f'}(1) \geq \frac{\lambda}{2}$.*

Proof. We recall a bit of notation: let $\mathcal{B}_f(1)$ denote the boundary of the coordinate 1 in the function f i.e. the family of all the sets $S \subseteq [2n] \setminus \{1\}$ such that $f(S) = 0, f(S \cup \{1\}) = 1$. For an integer $j \in \{0,1,\ldots,2n-1\}$, let $\mu_f(j)^{(1)}$ denote the fraction of subsets of $[2n] \setminus \{1\}$ of size j that are in $\mathcal{B}_f(1)$. For ease of notation, we let $\mu(j) = \mu_f(j)^{(1)}$, and $\mu'(j) = \mu_{f'}(j)^{(1)}$. Consider a set $S \subseteq [2n] \setminus \{1\}$ such that $S \in \mathcal{B}_f(1)$. Note that

$$S' = \{i-1 : i > 2, i \in S\}$$

satisfies $S' \in \mathcal{B}_{f'}(1)$. Suppose that $S_1, S_2 \in \mathcal{B}_f(1)$ such that $|S_1| = |S_2| = j$, $S_1 \neq S_2$ and $2 \notin S_1 \cup S_2$. Then, the above definition satisfies $S_1' \neq S_2'$, $S_1', S_2' \in \mathcal{B}_{f'}(1)$ and $|S_1'| = |S_2'| = j$. This implies that

$$\Big|\{S : S \in \mathcal{B}_f(1), |S| = j, 2 \notin S\}\Big| \leq \left|\mathcal{B}_{f'}(1) \cap \binom{[2n-1] \setminus \{1\}}{j}\right|$$

Similarly,

$$\Big|\{S : S \in \mathcal{B}_f(1), |S| = j, 2 \in S\}\Big| \leq \left|\mathcal{B}_{f'}(1) \cap \binom{[2n-1] \setminus \{1\}}{j-1}\right|$$

Summing the two, we obtain that

$$\Big|\{S : S \in \mathcal{B}_f(1), |S| = j\}\Big| \leq \left|\mathcal{B}_{f'}(1) \cap \binom{[2n-1] \setminus \{1\}}{j-1}\right| + \left|\mathcal{B}_{f'}(1) \cap \binom{[2n-1] \setminus \{1\}}{j}\right|$$

We can rewrite it as

$$\binom{2n-1}{j}\mu(j) \leq \binom{2n-2}{j}\mu'(j) + \binom{2n-2}{j-1}\mu'(j-1) \, \forall j \in [2n-2]$$

As $\binom{2n-1}{j} = \binom{2n-2}{j} + \binom{2n-2}{j-1}$ for every $j \in [2n-2]$, we get that

$$\mu(j) \leq \mu'(j) + \mu'(j-1)$$

for all $j \in [2n-2]$. Also note that $\mu(0) \leq \mu'(0)$, and $\mu(2n-1) \leq \mu'(2n-2)$. Summing over all these inequalities, we get that

$$\sum_{j\in\{0,1,\ldots,2n-2\}} \mu'(j) \geq \frac{1}{2} \sum_{j\in\{0,1,\ldots,2n-1\}} \mu(j) \geq \frac{\lambda(2n)}{2} = n\lambda$$

Thus,

$$\Phi_{f'}(1) = \frac{\sum_{j\in\{0,1,\ldots,2n-2\}} \mu'(j)}{2n-1} \geq \frac{\lambda}{2} \, . \qquad \square$$

Before proving the second step, we prove the following key lemma regarding the distribution of the boundary subsets.

Lemma 19. *Let $f' : \{0,1\}^{2n-1} \to \{0,1\}$ be a monotone Boolean function such that $\Phi_{f'}(1) = \lambda$ with $\lambda \geq \frac{1}{n}$. For an integer $j \in \{0,1,\ldots,2n-2\}$, let $\mu'(j) = \mu_{f'}(j)^{(1)}$. Then, there exists an absolute constant $\gamma := \gamma(\lambda) > 0$ such that*

$$\frac{\sum_{j=0}^{n-1} \mu'(2j)}{n} \geq \gamma$$

Proof. We prove that there exist constants (depending on λ) $c_1 < c_2, c > 0$ such that for all j such that $c_1 n \leq j \leq c_2 n$, we have $\mu'(j) \geq c$, and $c_2 - c_1 \geq \frac{\lambda^2}{2}$. This directly implies the required claim with $\gamma = \Omega(c\lambda^2)$.

For a pair of integers $0 \leq i < j \leq 2n-2$, we define the following parameter $\mu'(i,j)$ as the fraction of the pair of subsets (S,T) where $S, T \subseteq \{2,3,\ldots,2n-1\}, |S| = i, |T| = j, S \subseteq T$ that satisfy $S \in \mathcal{B}_{f'}(1), T \in \mathcal{B}_{f'}(1)$.

$$\mu'(i,j) = \frac{|\{(S,T) : |S| = i, |T| = j, S \subseteq T, S \in \mathcal{B}_{f'}(1), T \in \mathcal{B}_{f'}(1)\}|}{\binom{2n-2}{i}\binom{2n-2-i}{j}}$$

We first claim that there exist constants (depending on λ) $c_1 < c_2, c > 0$ such that $\mu'(c_1 n, c_2 n) \geq c$, and $c_2 - c_1 \geq \frac{\lambda^2}{2}$. Consider a uniformly random permutation of $[2n-1] \setminus \{1\}$ denoted by $\sigma = (\sigma(1), \sigma(2), \ldots, \sigma(2n-2))$. For an integer $j \in \{0,1,\ldots,2n-2\}$, let S_j be the random variable that is the union of the prefix of σ containing the first j elements.

$$S_j := \{\sigma(1), \sigma(2), \ldots, \sigma(j)\}, \, \forall j \in \{0,1,\ldots,2n-2\}.$$

For each $j \in \{0,1,\ldots,2n-2\}$, the subset S_j is uniformly distributed in $\binom{[2n-1]\setminus\{1\}}{j}$. For $j \in \{0,1,\ldots,2n-2\}$, let X_j be the indicator random variable for the event that $S_j \in \mathcal{B}_{f'}(1)$. By the definition of $\mu'(j)$, we get

$$\mathbb{E}[X_j] = \mu'(j) \ \forall j \in \{0,1,\ldots,2n-2\}.$$

Let $X = X_0 + X_1 + \ldots + X_{2n-2}$ be the number of subsets in the set family ($\phi = S_0 \subset S_1 \subset S_2 \ldots \subset S_{2n-2} = [2n-1] \setminus \{1\}$) that are in $\mathcal{B}_{f'}(1)$. Using Equation (3.1), we get

$$\mathbb{E}[X] = \lambda(2n-1).$$

Using Jensen's inequality, we get that

$$\mathbb{E}\left[\binom{X}{2}\right] \geq \binom{\lambda(2n-1)}{2} = \frac{1}{2} \cdot \lambda(2n-1)\left(\frac{\lambda}{2}(2n-2) + n\lambda - 1\right) \geq \frac{\lambda^2}{2}\binom{2n-1}{2}$$

wherein the final inequality, we used the fact that $\lambda n \geq 1$. Note that for every $i < j$, the marginal distribution of (S_i, S_j) is the uniform distribution over all the pairs of subsets (S, T) where $S, T \subseteq \{2, 3, \ldots, 2n-1\}, |S| = i, |T| = j, S \subseteq T$. Thus, by the definition of $\mu'(i,j)$, we get that $\mu'(i,j) = \mathbb{E}[X_i X_j]$, for $0 \leq i < j \leq 2n-2$. Therefore we have

$$\mathbb{E}\left[\binom{X}{2}\right] = \mathbb{E}\left[\sum_{0 \leq i < j \leq 2n-2} X_i X_j\right] = \sum_{0 \leq i < j \leq 2n-2} \mathbb{E}[X_i X_j] = \sum_{0 \leq i < j \leq 2n-2} \mu'(i,j)$$

Thus,

$$\sum_{0 \leq i < j \leq 2n-2} \mu'(i,j) \geq \frac{\lambda^2}{2}\binom{2n-1}{2}$$

This implies that the expected value (over i, j) of $\mu'(i,j)$ is at least $\frac{\lambda^2}{2}$. Thus, with probability (over i, j) at least $\frac{\lambda^2}{4}$, we have $\mu'(i,j) \geq \frac{\lambda^2}{4}$. Hence, there exist integers p, q such that $q - p \geq \frac{\lambda^2}{8} n$ and $\mu'(p,q) \geq \frac{\lambda^2}{4}$, which proves the required claim with $p = c_1 n, q = c_2 n, c = \frac{\lambda^2}{4}$.

Next, we claim that $\mu'(j) \geq c$ for all j such that $c_1 n \leq j \leq c_2 n$. Fix an integer j with $c_1 n \leq j \leq c_2 n$. Consider a uniformly random sequence of subsets $S_1 \subseteq S_2 \subseteq S_3 \subseteq [2n-1] \setminus \{1\}$ such that $|S_1| = c_1 n, |S_2| = j, |S_3| = c_2 n$. The probability that $S_1 \in \mathcal{B}_{f'}(1), S_3 \in \mathcal{B}_{f'}(1)$ is equal to $\mu'(c_1 n, c_2 n)$ which is at least $\frac{\lambda^2}{4}$. Thus, using Proposition 12, with probability at least $\frac{\lambda^2}{4}$, $S_2 \in \mathcal{B}_{f'}(1)$. Note that the distribution of S_2 is uniform in $\binom{[2n-1]\setminus\{1\}}{j}$, and thus, we have $\mu'(j) \geq \frac{\lambda^2}{4}$. □

We now prove the second step in the proof.

Lemma 20. *Suppose that $f' : \{0,1\}^{2n-1}$ is a monotone Boolean function such that $\Phi_{f'}(1) \geq \lambda$ with $\lambda \geq \frac{1}{n}$. Let g be a random minor of f' with respect to $\pi_2 : [2n-1] \to [n]$ which is obtained by setting $\pi_2(1) = 1$, and for every $i > 1$, we randomly choose $j_1, j_2 \in [2n-1] \setminus \{1\}$ (without replacements) and set $\pi_2(j_1) = \pi_2(j_2) = i$. In other words, we choose a uniformly random partition of $[2n-1] \setminus \{1\}$ into $n-1$ pairs $P_2, P_3, \ldots, P_n$ and set $\pi_2(j) = i \, \forall j \in P_i$. Then, there exists $\gamma := \gamma(\lambda) > 0$ such that*

$$\mathbb{E}_{\pi_2}[\Phi_g(1)] \geq \gamma \,.$$

Proof. For ease of notation, we let $\mu'(j) = \mu_{f'}(j)^{(1)}$ and $\mu_g(j) = \mu_g(j)^{(1)}$. For a set $S \subseteq [n] \setminus \{1\}$ and a function $\pi_2 : [2n-1] \to [n]$ with $\pi_2(1) = 1$, and $|\pi_2^{-1}(i)| = 2$ for all $i \in \{2, 3, \ldots, n\}$, let $\pi_2^{-1}(S)$ be the $2|S|$ sized subset of $\{2, 3, \ldots, 2n-1\}$ defined as follows:

$$\pi_2^{-1}(S) := \{\pi_2^{-1}(i) : i \in S\}$$

For every set $S \subseteq \{2, 3, \ldots, n\}$, when $\pi_2 : [2n-1] \to [n]$ is a uniformly random 2-to-1 minor with $\pi_2(1) = 1$, and the rest $2n-2$ elements are partitioned into $n-1$ pairs uniformly at random, the set $\pi_2^{-1}(S)$ is distributed uniformly in $\binom{[2n-1]\setminus\{1\}}{2|S|}$. Also note that $S \in \mathcal{B}_g(1)$ if and only if $\pi^{-1}(S) \in \mathcal{B}_{f'}(1)$. Thus, for every set $S \subseteq \{2, 3, \ldots, n\}$, the probability that $S \in \mathcal{B}_g(1)$ (over the choice of π_2) is equal to $\mu'(2|S|)$. Summing over all such sets of size j, we get that for every $j \in \{0, 1, \ldots, n-1\}$, the expected value of $\mu_g(j)$ is equal to $\mu'(2j)$.

$$\mathbb{E}_{\pi_2}[\mu_g(j)] = \mu'(2j) \, \forall j \in \{0, 1, \ldots, n-1\}$$

By using Lemma 19, we can infer that there exists $\gamma = \gamma(\lambda) > 0$ such that $\sum_{j=0}^{n-1} \mathbb{E}_{\pi_2}[\mu_g(j)] = \sum_{j=0}^{n-1} \mu'(2j) \geq \gamma n$. Using Equation (3.1), we get

$$\mathbb{E}_{\pi_2}[\Phi_g(1)] = \mathbb{E}_{\pi_2}\left[\frac{\sum_{j=0}^{n-1} \mu_g(j)}{n}\right] = \frac{\sum_{j=0}^{n-1} \mathbb{E}_{\pi_2}[\mu_g(j)]}{n} \geq \gamma. \quad \square$$

Lemma 18 and Lemma 20 together prove that Shapley value behaves well under uniformly random 2-to-1 minors for monotone Boolean functions.

Lemma 21. *Suppose that $f : \{0,1\}^{2n} \to \{0,1\}$ is a monotone Boolean function such that $\Phi_f(1) \geq \lambda$ for some absolute constant $\lambda > 0$ with $\lambda \geq \frac{1}{n}$. Then, there exists $\gamma := \gamma(\lambda) > 0$ such that*

$$\mathbb{E}_\pi[\Phi_g(\pi(1))] \geq \gamma$$

where g is a minor of f with respect to the uniformly random 2-to-1 function π.

Proof. Combining Lemma 18 and Lemma 20, we can conclude that for every $i \in [2n], i > 1$, when $\pi : [2n] \to [n]$ is a uniformly random 2-to-1 minor conditioned on the fact that $\pi(1) = \pi(i)$, we have $\mathbb{E}_\pi[\Phi_g(\pi(1))] \geq \gamma$. Taking average over all the $i \in [2n], i > 1$, we get a proof that the same inequality holds when π is a uniformly random 2-to-1 minor. $\square$

3.4.2 Reduction

In his celebrated work proposing the Unique Games Conjecture [Kho02a], Khot also proposed the "2-to-1 conjecture" that the strong hardness of Label Cover holds when all the constraints of the Label Cover are 2-to-1 functions. The imperfect completeness version of this conjecture was recently established in a striking sequence of works [KMS17; Din+18a; Din+18b; KMS18]. Braverman, Khot, and Minzer [BKM21] put forth a stronger conjecture that states that the hardness of Label Cover holds when the distribution of 2-to-1 functions on edges incident on every vertex $u \in L$ is uniform over $\mathcal{F}_{2\to 1}$.

Definition 22. *(Rich 2-to-1 Label Cover instances) We call a Label Cover instance $\mathcal{G} = (G, \Sigma_L, \Sigma_R, \Pi)$ with $G = (L \cup R, E)$ being a bipartite graph*[3] *a rich 2-to-1 instance if the following hold.*

[3] In the definition of the Label Cover problem in Chapter 2, we have assigned the same alphabet to all the vertices in a Label Cover instance. Here, in the context of bipartite Label Cover instances, we allow different alphabets to the left and right sides of the graph.

1. *There exists an integer Σ such that $\Sigma_L = [2\Sigma]$, $\Sigma_R = [\Sigma]$, and every projection constraint Π_e, $e \in E$ is a 2-to-1 function.*
2. *For every vertex $u \in L$, the distribution of 2-to-1 functions $\mathcal{P}_u$ obtained by first sampling a uniformly random neighbor v of u, and then picking $\Pi_e, e = (u, v)$, is uniform over $\mathcal{F}_{2\to 1}(\Sigma)$.*

Conjecture 23. *(Rich 2-to-1 Conjecture with Perfect Completeness) [BKM21] For every $\epsilon > 0$, there exists an integer $\Sigma = \Sigma(\epsilon)$ such that given a rich 2-to-1 Label Cover instance $\mathcal{G}$, it is NP-Hard to distinguish between the following.*

1. *There is a labeling that satisfies all the constraints of $\mathcal{G}$.*
2. *No labeling can satisfy more than ϵ fraction of the constraints of $\mathcal{G}$.*

We are now ready to state the hardness part of our dichotomy. It is proved using the Label Cover-Long Code framework. This reduction is standard in the PCSP literature, see e.g., [Bar+21].

Theorem 24. *Assume the Rich 2-to-1 Conjecture. Let PCSP(Γ) be a Boolean Ordered PCSP such that there exists an absolute constant $\lambda > 0$ with $\max_{i\in[n]} \Phi_f(i) \geq \lambda$ for all functions $f : \{0,1\}^n \to \{0,1\}$, $f \in \mathrm{Pol}(\Gamma)$. Then PCSP$(\Gamma)$ is NP-Hard.*

Proof. Let $\Gamma = \{(A_1, B_1), (A_2, B_2), \ldots, (A_l, B_l)\}$ be the PCSP under consideration, where each A_i is a subset of $\{0,1\}^{k_i}$ for all $i \in [l]$, and similarly, each B_i is a subset of $\{0,1\}^{k_i}$ for all $i \in [l]$. We start from a rich 2-to-1 Label Cover instance $\mathcal{G} = (G, [2\Sigma], [\Sigma], \Pi)$ with $G = (L \cup R, E)$. For ease of notation, we use Σ_w to denote 2Σ if $w \in L$, and Σ if $w \in R$. For every vertex $w \in L \cup R$, we have a set of 2^{Σ_w} nodes denoted by $L_w = \{w\} \times \{0,1\}^{\Sigma_w}$ referred to as the long code corresponding to w. The elements of our output PCSP instance V is the union of all the long code nodes.

$$V = \bigcup_{w\in L\cup R} L_w$$

We add two types of constraints.

1. Polymorphism Constraints. For every $i \in [l]$, we add the following constraints using the pair of predicates (A_i, B_i). For every $w \in L \cup R$, and multiset of vectors $\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^{k_i} \in \{0,1\}^{\Sigma_w}$ satisfying
$$(\mathbf{x}^1_j, \mathbf{x}^2_j, \ldots, \mathbf{x}^{k_i}_j) \in A_i \,\forall j \in [\Sigma_w],$$
we add the constraint on the k_i nodes $\{w, \mathbf{x}^1\}, \{w, \mathbf{x}^2\}, \ldots, \{w, \mathbf{x}^{k_i}\}$.
2. Equality Constraints. For every edge $e = (u, v)$ of the Label Cover instance with the constraint $\Pi_e : [2\Sigma] \to [\Sigma]$, we add the following set of equality constraints. For every $\mathbf{x} \in \{0,1\}^{2\Sigma}$ and $\mathbf{y} \in \{0,1\}^{\Sigma}$ such that for all $j \in [2\Sigma]$, $\mathbf{x}_j = \mathbf{y}_{\Pi_e(j)}$, we add an equality constraint between $\{u, \mathbf{x}\}$ and $\{v, \mathbf{y}\}$ ensuring that the two nodes are assigned the same value. The fact that we can add the equality constraints follows either by identifying the variables together, or by observing that the polymorphism minion of any PCSP remains the same when we add the equality predicate (see Chapter 5).

Completeness. Suppose that there exists a labeling σ that satisfies all the constraints of the Label Cover instance. For every node $\{w, \mathbf{x}\} \in V$, we assign the dictator function $\mathbf{x}_{\sigma(w)} \in \{0,1\}$. By the way we have added the polymorphism constraints, any dictator assignment satisfies them. The equality constraints are also satisfied as the labeling satisfies all the constraints of $\mathcal{G}$.

Soundness. Suppose that there exists an assignment $f : V \to \{0,1\}$ that satisfies all the polymorphism constraints and the equality constraints. Then, we claim that there exists a labeling σ that satisfies $\epsilon := \epsilon(\lambda) > 0$ fraction of the constraints of the Label Cover instance $\mathcal{G}$.

For a vertex $w \in L \cup R$, let $f_w : \{0,1\}^{\Sigma_w} \to \{0,1\}$ denote the function f restricted to L_w. Note that f_w is a polymorphism of the PCSP Γ for all $w \in L \cup R$. As every polymorphism of Γ has a coordinate with Shapley value at least λ, for every $u \in L$, we define the set $S(u)$ that is non-empty as follows:

$$S(u) = \{i \in [2\Sigma] : \Phi_{f_u}(i) \geq \lambda\}$$

As $\sum_{i\in[n]} \Phi_f(i) = 1$ for all functions $f : \{0,1\}^n \to \{0,1\}$, we have $|S(u)| \leq \frac{1}{\lambda}$ for all $u \in L$.

As a corollary of Lemma 21, we can conclude that there exists $\gamma = \gamma(\lambda) > 0$ such that for every monotone Boolean function $f : \{0,1\}^{2\Sigma} \to \{0,1\}$ with $\Phi_f(i) \geq \lambda$, when g is a minor of f with respect to a uniformly random 2-to-1 function $\pi : [2\Sigma] \to [\Sigma]$, $\Phi_g(\pi(1)) \geq \frac{\gamma}{2}$ with probability at least $\frac{\gamma}{2}$. Note that applying Lemma 21 requires that $\lambda \geq \frac{1}{\Sigma}$. However, even when $\lambda < \frac{1}{\Sigma}$, by picking the coordinate with the largest Shapley value, we can still assume that in every long code function, there is a coordinate with Shapley value at least $\frac{1}{2\Sigma} = \Theta(\lambda)$, and then apply Lemma 21. Using this γ, for every $v \in R$, we define the set $S(v)$ as

$$S(v) = \left\{i \in [\Sigma] : \Phi_{f_v}(i) \geq \frac{\gamma}{2}\right\}$$

By definition, we have $|S(v)| \leq \frac{2}{\gamma}$ for all $v \in R$. As the Label Cover instance is rich 2-to-1, for every $u \in L$, when we pick a uniformly random edge $e = (u, v)$ adjacent to u with constraint $\Pi_e : [2\Sigma] \to [\Sigma]$, with probability at least $\frac{\gamma}{2}$, there exist $i_1 \in [2\Sigma], i_2 \in [\Sigma]$ such that $\Phi_{f_u}(i_1) \geq \lambda$, $\Phi_{f_v}(i_2) \geq \frac{\gamma}{2}$, and $\Pi_e(i_1) = i_2$.

We now pick a labeling σ of $\mathcal{G}$ by picking uniformly random label from $S(w)$ for all $w \in L\cup R$. By the above argument, for every $u \in L$, the expected number of constraints of $\mathcal{G}$ that are adjacent to u that the labeling σ satisfies is at least $\frac{\gamma}{2} \cdot \lambda\frac{\gamma}{2}$. Summing over all $u \in L$, σ satisfies at least $\Omega(\lambda\gamma^2)$ fraction of the constraints of $\mathcal{G}$ in expectation. Thus, there exists a labeling to $\mathcal{G}$ that satisfies $\epsilon = \Omega(\lambda\gamma^2) > 0$ fraction of the constraints, which completes the proof. $\square$

3.5 Adversarial 2-to-1 minor

We construct an example of a 2-to-1 minor where the Shapley value alters completely after taking the minor.

Theorem 25. *Let $n \geq 2$ be a positive integer. There exist two monotone Boolean functions $f : \{0,1\}^{2n} \to \{0,1\}$ and $g : \{0,1\}^n \to \{0,1\}$ such that g is a 2-to-1 minor of f with respect to the 2-to-1 function $\pi : [2n] \to [n]$ defined as $\pi(i) = \lceil \frac{i}{2} \rceil$. Furthermore,*

1. *$\Phi_g(1) = \Omega(1)$, and $\Phi_g(j) = o(1)$ for all $j > 1$.*
2. *$\Phi_f(3) = \Omega(1)$, and $\Phi_f(i) = o(1)$ for all $i \in [2n], i \neq 3$.*

Proof. Similar to the proof of Theorem 24, we construct the minor function pair in two steps.

1. First, we construct Boolean monotone functions $f : \{0,1\}^{2n-1} \to \{0,1\}$ and $g : \{0,1\}^n \to \{0,1\}$ such that g is a minor of f with respect to the function $\pi : [2n-1] \to [n]$ defined as $\pi(1) = 1, \pi(i) = \lceil \frac{i+1}{2} \rceil$ for all $i > 1$. Furthermore, $\Phi_g(1) = \Omega(1)$, and $\Phi_g(j) = o(1)$ for all $j > 1$. We also have $\Phi_f(2) = \Omega(1)$, and $\Phi_f(i) = o(1)$ for all $i \in [2n-1], i \neq 2$.
2. We define the function $f' : \{0,1\}^{2n} \to \{0,1\}$ as

$$f'(y_1, y_2, \dots, y_{2n}) = f(y_1, y_3, \dots, y_{2n})$$

   Note that g is a minor of f' with respect to the 2-to-1 function $\pi : [2n] \to [n]$ defined as $\pi(i) = \lceil \frac{i}{2} \rceil$. Furthermore, by definition, we have $\Phi_{f'}(3) = \Omega(1)$, and $\Phi_{f'}(i) = o(1)$ for all $i \in [2n], i \neq 3$.

Henceforth, our goal is to construct a pair of functions as in the first step above.

We define a partial Boolean function to be a function from $\{0,1\}^n \to \{0,1,?\}$. A partial Boolean function on n variables is monotone if for all $\mathbf{p} \in \{0,1\}^n$ and $\mathbf{q} \in \{0,1\}^n$ such that $\mathbf{p} \leq \mathbf{q}$, if $f(\mathbf{p}) = 1$, then $f(\mathbf{q}) = 1$, and if $f(\mathbf{q}) = 0$, then $f(\mathbf{p}) = 0$.

Now, consider $g : \{0,1\}^n \to \{0,1\}$ to be

$$g(\mathbf{x}) = \begin{cases} 1 \text{ if } \sum_{j=2}^{n} x_j \geq \frac{51n}{100} \\ 0 \text{ if } \sum_{j=2}^{n} x_j \leq \frac{49n}{100} \\ x_1 \text{ if } \frac{49n}{100} < \sum_{j=2}^{n} x_j < \frac{51n}{100} \end{cases}$$

By definition, g is a monotone function, and using Equation (3.1), we can infer that $\Phi_g(1) = \frac{1}{50}$, and $\Phi_g(j) < \frac{1}{n}$ for all $j > 1$.

We now construct f in three steps. Start with $f = '?'$.

1. (Preserving the minor) First, set the value of entries of f that are of the form $(x_1, x_2, x_2, \cdots, x_n, x_n)$ as

$$f(x_1, x_2, x_2, \dots, x_n, x_n) = g(x_1, x_2, \dots, x_n) \ \forall \mathbf{x} \in \{0,1\}^n$$

   We then extend it both upwards and downwards i.e. if $f(\mathbf{p})$ is set to 1 and $\mathbf{p} \leq \mathbf{q}$, then set $f(\mathbf{q}) = 1$ as well, and similarly, if $f(\mathbf{q})$ is set to 0, and $\mathbf{p} \leq \mathbf{q}$, then we set $f(\mathbf{p}) = 0$. This ensures that g is a minor of f and that the partial function f is monotone.
2. (Destroying the influence of 1) Next, we ensure that the Shapley value of the coordinate 1 is low by the following operation: consider all $\mathbf{y}$ such that $f(\mathbf{y})$ has not been set in the first step, $y_1 = 0$ and $f(1, y_2, \cdots, y_{2n-1})$ is already set to 1 in the first step. Then set $f(\mathbf{y})$ to be 1. Similarly, if $\mathbf{y}$ satisfies $y_1 = 1$ and $f(0, y_2, \cdots, y_{2n-1})$ is already set to 0 in the first step, set $f(\mathbf{y})$ to be 0 if it has not been set in the first step.
   We claim that the updated partial function f is still a monotone partial function. Consider $\mathbf{p}, \mathbf{q} \in \{0,1\}^{2n-1}$ such that $\mathbf{p} \leq \mathbf{q}$. Suppose that $f(\mathbf{p})$ is set to be 1. If it is set in the first step, as we extended the partial function upwards in the first step, $f(\mathbf{q}) = 1$ as well. If

$f(\mathbf{p})$ is set to be 1 in the second step, it implies that $f(\mathbf{p}')$ has been set to 1 in the first step, where $\mathbf{p}'$ is obtained from $\mathbf{p}$ by setting p_1 to be 1. Let $\mathbf{q}' \in \{0,1\}^{2n-1}$ be obtained from $\mathbf{q}$ by setting $q_1 = 1$. As $\mathbf{p}' \leq \mathbf{q}'$, $f(\mathbf{q}')$ has been set to 1 in the first step as well. Thus, $f(\mathbf{q})$ is set to be 1 in the second step. The same argument can be used to show that if $f(\mathbf{q}) = 0$, then $f(\mathbf{p}) = 0$ as well.

3. (Adding influence to 2) For all $\mathbf{y}$ for which $f(\mathbf{y}) = '?'$ set $f(\mathbf{y}) = y_2$. The fact that the final function f is monotone follows from observing that any completion of a partial monotone function using a monotone function results in a monotone function.

Finally, our goal is to argue about the Shapley value of the coordinates of the function f. First, we show that the Shapley value of the coordinate 1 in f is $o(1)$. Suppose there exists $\mathbf{p} = (0, y_2, y_3, \cdots, y_{2n-1})$ and $\mathbf{q} = (1, y_2, y_3, \cdots, y_{2n-1})$ such that $f(\mathbf{p}) = 0$ and $f(\mathbf{q}) = 1$. We claim that both the values $f(\mathbf{p})$ and $f(\mathbf{q})$ are set in the first step of the above procedure. Suppose for contradiction that this is not the case. If neither of them is set in the first step, then they will not be set in the second step either, and in the third step, both of them will be assigned the same value, a contradiction. If exactly one of them is set in the first step, then in the second step, the other value would be set to be equal to it, a contradiction as well. Thus, both the values $f(\mathbf{p})$ and $f(\mathbf{q})$ are set in the first step.

Let $B = \mathcal{B}_g(1) \subseteq \{0,1\}^{n-1}$ be the boundary of the coordinate 1 in g. As $f(\mathbf{q})$ is set to be 1 in the first step, there exists $\mathbf{x} \in \{0,1\}^n$ such that $g(\mathbf{x}) = 1$ and $(x_1, x_2, x_2, \cdots, x_n, x_n) \leq \mathbf{q}$. As $\mathbf{x}$ is not less than or equal to $\mathbf{p}$, we can conclude that $x_1 = 1$ and $g(0, x_2, x_3, \cdots, x_n) = 0$. In other words, $(x_2, x_3, \cdots, x_n) \in B$. Similarly, there exists $\mathbf{x}'$ such that $g(\mathbf{x}') = 0$ and $(x'_1, x'_2, x'_2, \cdots, x'_n, x'_n) \geq \mathbf{p}$. By the same argument as above, we can conclude that $(x'_2, x'_3, \cdots, x'_n) \in B$. Combining the both, we can conclude that there exist $\mathbf{x}, \mathbf{x}' \in B$ such that $(x_2, x_2, x_3, x_3, \ldots, x_n, x_n) \leq (y_2, y_3, \cdots, y_{2n-2}) \leq (x'_2, x'_2, x'_3, x'_3, \ldots, x'_n, x'_n)$. Note that if the above inequality is true for a $(y_2, y_3, \cdots, y_{2n-2})$, we directly get that $(y_2, y_3, \cdots, y_{2n-2})$ is in the boundary of the coordinate 1 in f.

Observe that the boundary of coordinate 1 in g is the set of vectors $(x_2, x_3, \cdots, x_n)$ such that $\frac{49}{100}n \leq \sum_{j=2}^{n} x_j \leq \frac{51}{100}n$. By the previous argument, we can deduce that the boundary $B' = \mathcal{B}_f(1)$ of the coordinate 1 in f is the set of vectors $\mathbf{y} = (y_2, y_3, \cdots, y_{2n-1})$ that satisfy the following property: The number of $i \in [n-1]$ such that both $y_{2i} = y_{2i+1} = 1$ is at least $\frac{49}{100}n$. Similarly, the number of $i \in [n-1]$ such that $y_{2i} = y_{2i+1} = 0$ is at least $\frac{49}{100}n$. Observe that this implies that we require that $\frac{49}{50}n \leq \sum_{j=2}^{2n-1} y_j \leq \frac{51}{50}n$. However, for every integer l such that $\frac{49}{50}n \leq l \leq \frac{51}{50}n$, when we sample a uniformly random vector $\mathbf{y} = (y_2, y_3, \ldots, y_{2n-1})$ with $\sum_{j=2}^{2n-1} y_j = l$, the probability that the number of $i \in [n-1]$ such that both $y_{2i} = y_{2i+1} = 1$ is at least $\frac{49}{100}n$ is $o(\frac{1}{n})$. Thus, using Equation (3.1), we can infer that the Shapley value of the coordinate 1 in f is $o(1)$.

We now show that the coordinate 2 has $\Omega(1)$ Shapley value in f. Consider $\mathbf{y} = (y_1, y_2, \ldots, y_{2n-1})$ such that $\frac{49n}{50} < \mathsf{hw}(\mathbf{y}) \leq \frac{51n}{50}$. If the number of i such that both $y_{2i} = y_{2i+1} = 1$ is less than $\frac{49}{100}n$, we have $(y_1, y_3, \ldots, y_{2n-1}) \in \mathcal{B}_f(2)$. However, for every integer l such that $\frac{49}{50}n \leq l \leq \frac{51}{50}n$, when we sample a uniformly random $\mathbf{y}$ with $\mathsf{hw}(\mathbf{y}) = l$, with probability $1 - o(1)$, the number of i such that both $y_{2i} = y_{2i+1} = 1$ is less than $\frac{49}{100}n$. Thus, using Equation (3.1), we can infer that the Shapley value of the coordinate 2 is $\Omega(1)$ in the function f. Finally, by symmetry, we can observe that $\Phi_f(i) = \Phi_f(3)$ for all $i \geq 3$, and thus, $\Phi_f(i) = o(1)$ for all $i \geq 3$. $\square$

Chapter 4

d-to-1 Hardness of Coloring 3-colorable graphs with $O(1)$ colors

4.1 Introduction

Determining if a graph is 3-colorable is one of the classic NP-complete problems. Thus, given a 3-colorable graph it is NP-hard to color it with 3 colors. The best known polynomial time algorithms for coloring 3-colorable graphs use about $n^{0.2}$ colors, where n is the number of vertices in the graph [KT17]. On the other hand, on the hardness front, we only know that 5-coloring 3-colorable graphs is NP-hard [Bar+21].

This embarrassingly large gap between the hardness and algorithmic results has prompted the quest for conditional hardness results for approximate graph coloring. The canonical starting point for most strong inapproximability results is the Label Cover problem. The Unique Games Conjecture of Khot [Kho02a], which asserts strong inapproximability of Label Cover when the constraint maps are bijections, has formed the basis of numerous tight hardness results for problems which have defied NP-hardness proofs. However, the imperfect completeness inherent in the Unique Games Conjecture makes it unsuitable as the basis for hardness results for graph coloring, where we want *all* edges to be properly colored under the coloring.

In [Kho02a], along with the Unique Games Conjecture, Khot introduced the d-to-1 conjecture. The d-to-1 conjecture says that given a Label Cover instance whose constraint relations are d-to-1 functions, it is NP-hard to decide if there exists a labelling that satisfies all the constraints or no labelling can satisfy even an ϵ fraction of constraints, for arbitrarily small $\epsilon > 0$. (The key is that d can be held fixed and achieve soundness $\epsilon \to 0$.) Constraints similar to 2-to-1 also played an implicit role in the beautiful work of Dinur and Safra on inapproximability of vertex cover [DS05].

Based on the 2-to-1 conjecture, Dinur, Mossel and Regev [DMR09], extending the invariance principle based techniques of [Kho+07; MOO10], proved the hardness of coloring graphs that are promised to be 4-colorable with any constant number of colors. Furthermore, they prove the same for 3-colorable graphs under a certain "fish shaped" variant of the 2-to-1 conjecture. In this work, we prove that the same result can be proved under the weaker assumption of d-to-1 conjecture[1], for some (arbitrarily large) constant d.

Theorem 26. *Assume that d-to-1 conjecture is true for some constant d. Then, for every positive integer $t \geq 3$, it is NP-hard to color a 3-colorable graph G with t colors.*

We stress that the d-to-1 conjecture insists on perfect completeness (i.e., hardness on satisfiable instances), and this important feature seems necessary for its implications for coloring problems, where we seek to properly color all edges. The variant of the 2-to-1 conjecture where one settles for near-perfect completeness was recently established in a striking sequence of works [KMS17; Din+18a; Din+18b; KMS18].

The result of [DMR09] in fact shows hardness of finding an independent set of density ϵ in a 3-colorable graph for arbitrary $\epsilon > 0$ (which immediately implies the hardness of finding a coloring with $1/\epsilon$ colors). Our result in Theorem 26 above does *not* get this stronger hardness for finding independent sets. But it is conditioned on the d-to-1 conjecture for arbitrary d rather than the specific 2-to-1 conjecture. We note that proving the d-to-1 conjecture for some large d could be significantly easier than the 2-to-1 conjecture, so Theorem 26 perhaps provides an avenue for resolving a longstanding challenge concerning the complexity of approximate graph coloring.

Our proof of Theorem 26 is a simple combination of two results. First, following the methodology of [DMR09], we prove that the d-to-1 conjecture implies that coloring a $2d$-colorable graph with $O(1)$ colors is NP-hard. The result of [DMR09] is the $d = 2$ case of this claim. In fact, they state in a future work section that the d-to-1 conjecture should imply hardness of $O(1)$-coloring q-colorable graphs for some large enough $q = q(d)$. However, they did not specify the details of the reduction or an explicit value of q, and mention determining the dependence of q on d as an interesting question. Here we show the conditional hardness based on d-to-1 conjecture holds for $q = 2d$ (achieving $q < 2d$ seems unlikely with the general reduction approach of [DMR09]).

The key technical ingredient necessary for such a reduction is a symmetric Markov chain on $[q]^d$ where transitions are allowed only between disjoint tuples and which has spectral radius bounded away from 1. We show the existence of such a symmetric Markov chain for $q = 2d$. We do so via a connection to matrix scaling, which enables us to deduce the necessary chain at a conceptual level without messy calculations. Specifically, we use the result [CD72], which follows from the Sinkhorn-Knopp iterative matrix scaling algorithm [SK67], that if a non-negative symmetric matrix A has *total support* then there is a symmetric doubly stochastic matrix supported on the non-zero entries of A. When A is the adjacency matrix of a graph G, the total support condition is equivalent to every edge of G belonging to a cycle cover. We describe a graph on $[q]^d$ whose edges connect disjoint tuples and where every edge belongs to a cycle cover.

Our second ingredient is a remarkable yet simple reduction due to Krokhin, Opršal, Wrochna and Živný [Kro+20], which exploits the relation between the arc-chromatic number and chromatic number of a digraph [PR81]. Let $b : \mathbb{N} \to \mathbb{N}$ be defined by $b(n) := \binom{n}{\lfloor n/2 \rfloor}$. Their result then is that $b(t)$-coloring $b(c)$-colorable graphs is polynomial time (in fact logspace) reducible to

[1] For d-to-1 Label Cover, there are two definitions possible, one where the constraint maps are *at most* d-to-1 with each element in the range having at most d pre-images, and one where the constraint maps are *exactly* d-to-1. In this chapter, we stick with the exact variant.

t-coloring c-colorable graphs. Since $b(n)$ is increasing and $b(n) > n$ for all $n \geq 4$, it follows that a NP-hardness result for $O(1)$-coloring q-colorable graphs also implies NP-hardness of $O(1)$-coloring 4-colorable graphs. Furthermore, the NP hardness of $O(1)$-coloring of 3-colorable graphs follows from the above by applying the arc graph reduction twice to K_4.

Overview. In Section 4.2, we define the Label Cover problem, and state the d-to-1 conjecture formally. We also introduce low degree influences that we need later. In Section 4.3, we first prove the existence of the Markov chain with required properties, and then describe the reduction from Label Cover to Approximate Coloring. We note that the reduction is in fact exactly the same one used in [DMR09], the difference being in using a different Markov Chain. For the sake of completeness, we present the reduction and the preliminaries required.

4.2 Preliminaries

We first formally define the hardness conjectures.

4.2.1 d-to-1 Conjecture

We first state the d-to-1 conjecture. As is the case with [DMR09], we will state and use the *exact* d-to-1 variant where the constraint maps have exactly d pre-images for each element in the range. Khot's original formulation only required that there are at most d pre-images for each element in the range. The d-to-1 conjecture becomes stronger for smaller d (so that the 2-to-1 is the strongest form of the conjecture)—this is obvious for the variant where the maps are at most d-to-1. For the exact variant, if we allow the Label cover graph to have multiple edges, we can reduce d-to-1 conjecture to $(d+1)$-to-1 conjecture using a simple argument. We present this reduction in Section 4.4. On that note, we remark without details that our reduction indeed works with the multigraph variant of d-to-1 conjecture.

Conjecture 27. *((Exact) d-to-1 Conjecture) For every $\epsilon > 0$, given a bipartite Label Cover instance $G = ((V = X \cup Y, E), (dR, R), \Psi)$ satisfying the following constraints:*

(i) We refer to X as the vertices on the left, and Y as the set of vertices on the right. The vertices belonging to X are to be assigned labels from $[dR]$ while the vertices in Y are to be assigned labels from $[R]$.

(ii) The constraints are d-to-1 i.e. for every $b \in [R]$, there are precisely d values $a \in [dR]$ such that $(a, b) \in \Psi_e$ for every relation Ψ_e in the instance.

It is NP-hard to distinguish between the following cases:

1. *There is a labeling that satisfies all the constraints in G.*
2. *No labeling can satisfy more than ϵ fraction of constraints in G.*

Similar to the d-to-1 constraints, one can consider d-to-d constraints in the Label Cover. In order to do so, we define the relation $d \leftrightarrow d$ on $[dR] \times [dR]$:

$$d \leftrightarrow d = \{(di - p + 1, di - q + 1) \mid 1 \leq i \leq R,\ 1 \leq p, q \leq d\}\ .$$

A constraint $\psi \subseteq [dR] \times [dR]$ is said to be d-to-d if there exist permutations π_1 and π_2 on $[dR]$ such that $(a, b) \in \psi$ iff $(\pi_1^{-1}(a), \pi_2^{-1}(b)) \in d \leftrightarrow d$.

In [DMR09], it is proved that Conjecture 27 implies the following conjecture.

Conjecture 28. *(d-to-d conjecture) For every $\epsilon > 0$ and every $t \in \mathbb{N}$, there exists $R \in \mathbb{N}$ such that given a Label Cover instance $G = ((V, E), dR, \Psi)$ where all the constraints are d-to-d, it is NP-hard to distinguish between the following cases:*

(i) $sat(G) = 1$*, or*

(ii) $isat_t(G) < \epsilon$

Here, $sat(G)$ denotes the maximum fraction of constraints satisfied by any labeling. Similarly, $isat(G)$ denotes the size of the largest set $S \subseteq V$ such that there exists a labeling that satisfies all the constraints induced on S. The value $isat_t(G)$ denotes the size of largest set $S \subseteq V$ such that there exists a labeling that assigns at most t labels to each vertex that satisfies all the constraints induced on S. A constraint between u, v is said to be satisfied by labeling assigning multiple labels to u and v if and only if there exists at least one pair of labels to u and v among the multiple labels that satisfy the constraint.

4.2.2 Low degree influences

Next, we define the low degree influences that we need later. We refer the reader to [DMR09] for a comprehensive treatment of the same.

Let $\alpha_0 = \mathbf{1}, \alpha_1, \ldots, \alpha_{q-1}$ be an orthonormal basis of $\mathbb{R}^q$. We can define the set of functions $\alpha_x : [q]^n \to \mathbb{R}, x \in [q]^n$ as $\alpha_x(y) = (\alpha_{x_1}(y_1), \alpha_{x_2}(y_2), \ldots, \alpha_{x_n}(y_n))$. Observe that these functions form a basis for the functions from $[q]^n$ to $\mathbb{R}$. Let $\hat{f}(\alpha_x) = \langle f, \alpha_x \rangle$, where we define the inner product between functions $f, g : [q]^n \to \mathbb{R}$ as $\langle f, g \rangle = q^{-n} \sum_{x \in [q]^n} f(x)g(x)$. We define the low degree influence of f as follows:

Definition 29. *For a function $f : [q]^n \to \mathbb{R}$, the degree k influence of the coordinate i is defined as follows:*

$$I_i^{\leq k}(f) = \sum_{x : x_i \neq 0, |x| \leq k} \hat{f}^2(\alpha_x)$$

Note that the above definition is independent of the basis $\alpha_0, \alpha_1, \ldots, \alpha_{q-1}$ that we start with, as long as $\alpha_0 = \mathbf{1}$. From the above definition, we can infer that for functions $f : [q]^n \to [0, 1]$, the sum of low degree influences is bounded by

$$\sum_i I_i^{\leq k}(f) \leq k$$

For a vector $x \in [q]^{dR}$, let $\overline{x} \in [q^d]^R$ be the corresponding element in $[q^d]^R$ i.e.

$$\overline{x} = ((x_1, x_2, \ldots, x_d), (x_{d+1}, x_{d+2}, \ldots, x_{2d}), \ldots, (x_{dR-d+1}, x_{dR-d+2}, \ldots, x_{dR}))$$

Similarly, for $y \in [q^d]^R$, let $\underline{y}$ denote the inverse of above operation. We can extend this notion to functions as well: For a function $f : [q]^{dR} \to \mathbb{R}$, let the function $\overline{f} : [q^d]^R \to \mathbb{R}$ be defined naturally by

$$\overline{f}(y) = f(\underline{y})$$

Similarly, for a function $f : [q^d]^R \to \mathbb{R}$, let $\underline{f} : [q]^{dR} \to \mathbb{R}$ be defined as $\underline{f}(x) = f(\overline{x})$.

We need the following lemma:

Lemma 30. *For any function $f : [q]^{dR} \to \mathbb{R}$ and any $k \in \mathbb{N}$ and $i \in [R]$,*

$$I_i^{\leq k}(\overline{f}) \leq \sum_{j=1}^{d} I_{di-d+j}^{\leq dk}(f)$$

Proof. Fix a basis α_x of functions from $[q]^{dR} \to \mathbb{R}$ as above. The functions $\alpha_{\overline{x}}$ form a basis for functions from $[q^d]^R \to \mathbb{R}$, where $\alpha_{\overline{x}}(\overline{y}) = \alpha_x(y)$. Note that $\hat{\overline{f}}(\alpha_{\overline{x}}) = \hat{f}(\alpha_x)$. Thus we get

$$\begin{aligned}
\sum_i I_i^{\leq k}(\overline{f}) &= \sum_{\overline{x}:\overline{x}_i \neq (0,0,\ldots,0), |\overline{x}| \leq k} \hat{\overline{f}}^2(\alpha_{\overline{x}}) = \sum_{\overline{x}:\overline{x}_i \neq (0,0,\ldots,0), |\overline{x}| \leq k} \hat{f}^2(\alpha_x) \\
&\leq \sum_{x:\overline{x}_i \neq (0,0,\ldots,0), |x| \leq dk} \hat{f}^2(\alpha_x) \\
&\leq \sum_{j=1}^{d} \sum_{x:x_{di-d+j} \neq 0, |x| \leq dk} \hat{f}^2(\alpha_x) \\
&= \sum_{j=1}^{d} I_{di-d+j}^{\leq dk}(f)
\end{aligned}$$

□

Using the invariance principle and Borell's inequality, [DMR09] prove the following:

Theorem 31. *Let q be a fixed integer, and T be a symmetric Markov chain on $[q]$ with $r(T) < 1$. Then for every $\epsilon > 0$, there exists a $\delta > 0$ and a positive integer k such that the following holds: For every $f, g : [q]^n \to [0,1]$ if $\mathbb{E}[f] > \epsilon, \mathbb{E}[g] > \epsilon$ and $\langle f, Tg \rangle = 0$, then*

$$\exists i \in [n] : I_i^{\leq k}(f) \geq \delta, I_i^{\leq k}(g) \geq \delta$$

where $r(T)$ denotes the second largest eigenvalue (in absolute value) of T.

4.3 d-to-1 hardness for 3-colorable graphs

In this section, we will prove Theorem 26.

4.3.1 Reducing chromatic number to 3

The following lemma is present in [Kro+20] based on a beautiful result concerning the arc-chromatic numbers of digraphs from [PR81].

Lemma 32. *(Theorem 1.8 of [Kro+20]) Suppose there exists $q \in \mathbb{N}$ such that $O(1)$ coloring q-colorable graphs is NP-hard. Then, $O(1)$ coloring 3-colorable graphs is NP hard.*

Let Graph-Coloring(t, c) denote the promise problem of distinguishing if a graph can be colored with c colors, or cannot even be colored with t colors. The statement is proved by presenting a reduction from Graph-Coloring$(b(t), b(c))$ to Graph-Coloring(t, c) in polynomial time, for the function $b(n) := \binom{n}{\lfloor n/2 \rfloor}$. The reduction works by constructing the arc-graph of the underlying graphs, and using the property of arc graphs that the chromatic number of the arc graph can be bounded precisely using the chromatic number of the original graph. Since b is an increasing function and $b(n) > n$ for all $n \geq 4$, setting $c = 4$ and t large enough proves the statement claimed in the lemma. The reduction from 4-colorable graphs to 3-colorable graphs is achieved by applying the arc graph construction twice recursively.

Thanks to Lemma 32, we can restrict ourselves to the weaker goal of proving that $O(1)$ coloring q-colorable graphs is NP-hard for some fixed constant q assuming Conjecture 27. In fact, following [DMR09], we prove a stronger statement showing hardness of finding independent sets of ϵ fraction of vertices for any $\epsilon > 0$. Combined with Lemma 32, this immediately gives us Theorem 26.

Theorem 33. *Suppose that Conjecture 28 is true for a constant d. Then, there exists a constant $q = q(d)$ such that for every $\epsilon > 0$, given a graph G, it is NP-hard to distinguish the following cases:*

1. *G can be colored with q colors.*
2. *G does not have any independent set of relative size ϵ.*

In fact, we can take $q = 2d$.

In the remainder of the section, we will prove Theorem 33. We next develop the main technical ingredient that we will plug into the reduction framework of [DMR09] to establish Theorem 33.

4.3.2 A symmetric Markov chain supported on disjoint tuples

A Markov chain T defined on a state space Ω is said to be symmetric if the transition matrix of T is symmetric, namely for all pairs of states $x, y \in \Omega$, the probability of transition from x to y is equal to the probability of transition from y to x. Symmetry of the Markov chain ensures that the uniform distribution is stationary which is essential when we compose the Label Cover-Long Code reduction with the Markov chain. We define the spectral radius $r(T)$ of a symmetric Markov chain as the second largest eigenvalue in absolute value of its transition probability matrix, i.e., if $1 = \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_q$ are the eigenvalues, then $r(T) = \max(|\lambda_2|, |\lambda_q|)$.

We now show the existence of a symmetric Markov Chain T on $[q]^d$ with $r(T) < 1$ if $d \geq 2, q \geq 2d$. Furthermore, there is a nonzero transition probability between two elements $x, y \in [q]^d$ only if the support of x and y are disjoint. In [DMR09], such a Markov Chain is shown to exist for the values $(q, d) = (3, 1), (4, 2)$.

Lemma 34. *Suppose that $q, d \in \mathbb{N}, q \geq 2d, d \geq 2$. There exists a symmetric Markov chain T on $[q]^d$ such that $r(T) < 1$. Furthermore, if the transition $\{x_1, x_2, \ldots, x_d\} \leftrightarrow \{y_1, y_2, \ldots, y_d\}$ has positive probability in T, then $\{x_1, x_2, \ldots, x_d\} \cap \{y_1, y_2, \ldots, y_d\} = \phi$.*

Proof. We first construct an undirected graph G on $[q]^d$ such that there is an edge between $x, y \in [q]^d$ only if the support of x and y are disjoint. We then use a matrix scaling algorithm to obtain a symmetric Markov chain T from the adjacency matrix of G. For the resulting Markov chain to have $r(T) < 1$, we need that the underlying graph G is connected, and is not bipartite. Furthermore, for the scaling algorithm to produce a valid Markov chain, we need that every edge of G is present in a cycle cover, where a cycle cover of a graph is a disjoint union of cycles that covers every vertex in the graph. Note that we allow trivial 2-cycles in a cycle cover, where we just take an edge twice.

We say that two multisets $x = (x_1, x_2, \ldots, x_d), y = (y_1, y_2, \ldots, y_d) \in [q]^d$ are of the same *type* if the following condition holds: for all pairs of indices $i, j \in [d]$, $x_i = x_j$ if and only if $y_i = y_j$ and $(x_i - x_j)(y_i - y_j) \geq 0$. Note that this is an equivalence relation, and thus each element $x \in [q]^d$ uniquely determines its type.

Consider the graph $G = (V, E)$ where the vertex set is $V = [q]^d$. We add two kinds of edges in this graph. We add an edge between every pair of $x, y \in [q]^d$ that are of the same type, and have disjoint support. Let the subset of $[q]^d$ of elements that are supported on single element be denoted by S, i.e.,

$$S = \{(1, 1, \ldots, 1), (2, 2, \ldots, 2), \ldots, (q, q, \ldots, q)\} .$$

We also add edges between x and y if their support is disjoint, and at least one of x and y belongs to S.

First, we claim that G is connected. This follows from the fact that the set of nodes in S are connected to each other, and every vertex in V is adjacent to at least one vertex in S. As $q \geq 4$, the graph is not bipartite (indeed S induces a q-clique). We will now prove that every edge in this graph is part of a cycle cover. Given an undirected graph on vertex set V, a cycle cover of it is a function $\sigma : V \to V$ that is bijective, and $\sigma(u) = v$ only when u and v are adjacent in the underlying graph.

Towards this, we first prove that for every edge in G between multisets of the same type, there is a cycle cover that uses that edge. For each type, consider the graph obtained by taking the vertices as multisets of that type, and with edges between two multisets of the same type if they are disjoint. Note that for every type, this graph is isomorphic to a Kneser graph $KG(q, k)$ (for some $k \leq d$), whose vertex set corresponds to k-element subsets of $[q]$ and there is an edge between two subsets if they are disjoint.

By symmetry across the subsets, we can infer that the Kneser graphs are regular. Note that every regular graph contains a cycle cover: For a regular graph H, consider a bipartite graph H' which contains a copy of H on both the left side L, and right side R. There is an edge between $x \in L, y \in R$ of H' if and only if x, y are adjacent in H. As H is a regular graph, H' is a regular bipartite graph, and thus, contains a perfect matching. This perfect matching in H' directly gives a cycle cover of H. Furthermore, as Kneser graphs are also vertex-transitive, every edge in these graphs is part of a cycle cover.

Next, we consider edges of G that are between multisets of different types i.e. edges between multisets x, y where exactly one of x and y is in S. Consider an edge between $s \in S$ and $x \in V \setminus S$. As $q \geq 2d$, every multiset in G is adjacent to at least one multiset of the same type. Let y be a multiset that is adjacent to x in G and is of the same type as x. Let $s' \in S$ be chosen such that it is adjacent to y in G. As S is a complete subgraph of G, s and s' are adjacent in G. From the previous argument about edges between multisets of the same type, we can infer that there is a cycle cover of G where y is mapped to x, and s is mapped to s'. We can modify this cycle cover by transforming it as follows - $(s \to x)$ can be made part of cycle cover by transforming $(s \to s'), (y \to x)$ to $(s \to x), (y \to s')$ and keeping rest of the cycle cover intact. Thus, we have proved that every edge of G is part of a cycle cover.

Let A denote the adjacency matrix of the above graph G. Using the Sinkhorn Knopp iterative algorithm, it is proved in [CD72] that if a non-negative symmetric matrix A has total support, then there exists a diagonal matrix D such that DAD is a doubly stochastic matrix. A square matrix $A = (a_{ij})$ of order n is said to have total support if $A \neq 0$, and for every nonzero entry a_{ij} of A, there exists a permutation σ of $[n]$ such that $\sigma(i) = j$ and for all $e \in [n], a_{e,\sigma(e)} \neq 0$. When the matrix A is an adjacency matrix of a graph G, the total support condition translates to the requirement that every edge in G is part of a cycle cover, a property we have already shown to hold for the graph G.

Thus, we can apply the above scaling result, and view the resulting matrix $B = DAD$ as the transition matrix of a Markov chain T. As A and D are symmetric, B is symmetric, i.e., T is symmetric. As A is connected and no principal diagonal element of D is zero, T is connected as well. Note that every nonzero element of A stays nonzero in T, and A is not bipartite. The above two facts combined ensure that the spectral radius $r(T)$ of T is strictly less than 1. We conclude that there exists a symmetric Markov chain T on state space $[q]^d$ that has both the properties: (i) $r(T) < 1$, and (ii) there is nonzero probability of transition between two multisets only when their support is disjoint. □

4.3.3 Proof of Theorem 33

Let d be the constant for which Conjecture 27 is true. Thus, Conjecture 28 is true for the same value d as well. Choose q, T from Lemma 34 such that T is a symmetric Markov chain on $[q]^d$ such that $r(T) < 1$.

We now reduce the given d-to-d Label Cover instance to the problem of finding independent sets in q-colorable graphs. To be precise, given a Label Cover instance $G = ((V, E), dR, \Psi)$, we output a graph $G' = (V', E')$ such that

1. Completeness: If G is satisfiable, G' can be colored with q colors.
2. Soundness: If $isat_t(G) < \epsilon'$, then G' does not have any independent set of size ϵ.

The parameters t and ϵ' will be set later.

Reduction. Our reduction follows the standard Label Cover Long Code paradigm, and in particular closely mirrors [DMR09]. We replace each vertex $w \in V$ of the Label Cover with a set f_w of $[q]^{dR}$ nodes, each corresponding to a vertex in G'. Consider an edge $e = (u, v)$ where Ψ_e is an associated constraint with permutations π_1, π_2 on $[dR]$ such that $(a, b) \in \Psi_e$ if and only if $(\pi_1^{-1}(a), \pi_2^{-1}(b)) \in d \leftrightarrow d$.

We add an edge between $(x_1, x_2, \ldots, x_{dR}) \in f_u$ and $(y_1, y_2, \ldots, y_{dR}) \in f_v$ to E' if and only if

$$\forall i \in [R], T((x_{\pi_1(di-d+1)}, x_{\pi_1(di-d+2)}, \ldots, x_{\pi_1(di)}) \leftrightarrow (y_{\pi_2(di-d+1)}, y_{\pi_2(di-d+2)}, \ldots, y_{\pi_2(di)})) > 0$$

Completeness. Suppose $\sigma : V \to [dR]$ be a labeling satisfying all the constraints of the Label Cover instance G. We color the node $(x_1, x_2, \ldots, x_{dR}) \in f_w$ with $x_{\sigma(w)} \in [q]$. We claim that this is a legit q-coloring of G'. Suppose that we added an edge between $x \in f_u$ and $y \in f_v$. Let x be colored with x_a and y be colored with y_b. As $(a, b) \in \Psi_{(u,v)}$, we have $(\pi_1^{-1}(a), \pi_2^{-1}(b)) \in d \leftrightarrow d$. Thus, there exist $i \in [R], 1 \le p, q \le d$ such that $a = \pi_1(di - d + p)$ and $b = \pi_2(di - d + q)$. As we have added an edge between $x \in f_u$ and $y \in f_v$, $x_a \neq y_b$ as the Markov chain T has nonzero probability only between two elements of $[q]^d$ with disjoint support. Thus, there exists a q-coloring of G' when G is satisfiable.

Soundness. We prove the contrapositive that if G' has an independent set of relative size ϵ, then there exists a labeling of G with $isat_t(G) \ge \epsilon'$. Let $S \subseteq V'$ be the largest independent set of G'. We know that $|S| \ge \epsilon |V'|$. This implies that in at least $\epsilon' = \frac{\epsilon}{2}$ fraction of the long code blocks, at least $\frac{\epsilon}{2}$ fraction of nodes belong to S. Let this subset of V be denoted by Z. Our goal is to show that there exists a small set of labels $\tau : Z \to 2^{[dR]}$ to which we can decode the vertices in Z such that all the constraints induced in Z are satisfied by τ.

For every vertex $w \in Z$, we define functions $g_w : [q]^{dR} \to \{0, 1\}$ to be the indicator functions of set S inside the long code blocks corresponding to w i.e. $g_w(x) = 1$ if and only if $x \in S$. Consider an edge $e = (u, v)$ corresponding to the constraint Ψ_e induced in Z. Let the functions $f : [q]^{dR} \to \{0, 1\}$ and $g : [q]^{dR} \to \{0, 1\}$ be defined such that $f(x^{\pi_1}) = g_u(x)$ and $g(y^{\pi_2}) = g_v(y)$, where π_1 and π_2 are the permutations underlying the relation Ψ_e i.e. $(a, b) \in \Psi_e$ if and only if $(\pi_1^{-1}(a), \pi_2^{-1}(b)) \in d \leftrightarrow d$.

We note that $\langle f, Tg \rangle$ is equal to zero. In other words, suppose that $x, y \in [q]^{dR}, x \in f_u, y \in f_v$ are such that

$$\forall i \in [R], T((x_{di-d+1}, x_{di-d+2}, \ldots, x_{di}) \leftrightarrow (y_{di-d+1}, y_{di-d+2}, \ldots, y_{di})) > 0. \tag{4.1}$$

Then, $f(x)g(y) = 0$. Suppose for contradiction that there exist $x, y \in [q]^{dR}$ satisfying the above condition, and $f(x) = g(y) = 1$. Let $x' \in f_u, y' \in f_v$ be such that $(x')^{\pi_1} = x, (y')^{\pi_2} = y$. We have $g_u(x') = g_v(y') = 1$. That is, both $x' \in f_u, y' \in f_v$ are in the independent set S. However, Equation (4.1) can be rewritten as the following:

$$\forall i \in [R], T((x'_{\pi_1(di-d+1)}), (x'_{\pi_1(di-d+2)}), \ldots, x'_{\pi_1(di)}) \leftrightarrow (y'_{\pi_2(di-d+1)}, y'_{\pi_2(di-d+2)}, \ldots, y'_{\pi_2(di)})) > 0. \tag{4.2}$$

Note that this is precisely the condition for adding edges in G'. Thus, Equation (4.2) implies that $x' \in f_u$ and $y' \in f_v$ are adjacent in E', and thus cannot both be part of the independent set S. This completes the proof that $\langle f, Tg \rangle = 0$.

Thus, $\langle \overline{f}, T\overline{g} \rangle$ is also equal to zero, where $\overline{f} : [q^d]^R \to \{0, 1\}$ and $\overline{g} : [q^d]^R \to \{0, 1\}$ are the corresponding functions in $[q^d]^R$ of f, g. From the definition of Z, $\mathbb{E}(\overline{f}) \ge \frac{\epsilon}{2}$ and $\mathbb{E}(\overline{g}) \ge \frac{\epsilon}{2}$. We apply Theorem 31 to $\overline{f}$ and $\overline{g}$ to deduce that there exists $i \in [R]$, a positive integer $k = k(\epsilon)$ and

$\delta = \delta(\epsilon)$ such that $I_i^{\leq k}(\overline{f}) \geq \delta$ and $I_i^{\leq k}(\overline{g}) \geq \delta$. This motivates us to define the label set of vertex $w \in Z$, $L(w)$ as the following -

$$L(w) := \{i \in [dR] : I_i^{\leq dk}(g_w) \geq \frac{\delta}{d}\}$$

As the sum of k degree influences of all variables is at most k, the size of $L(w)$ is upper bounded by $\frac{kd}{\delta}$ for every v. Thus, we set the parameter t to be $\frac{kd}{\delta}$.

The final step is to prove that the labeling L is indeed a valid labeling inside edges induced in Z. Consider an edge $e = (u, v)$ induced in Z with the constraint relation being Ψ_e such that $(a, b) \in \Psi_e$ if and only if $(\pi_1(a), \pi_2(b)) \in d \leftrightarrow d$. Our goal is to show that there exist indices $\sigma_1, \sigma_2 \in [dR]$ such that $\sigma_1 \in L(u), \sigma_2 \in L(v)$ and $(\sigma_1, \sigma_2) \in \Psi_e$. Using Theorem 31, we can deduce that there exists $i \in [R]$ such that $I_i^{\leq k}(\overline{f}) \geq \delta$ and $I_i^{\geq k}(\overline{g}) \geq \delta$. Using Lemma 30, we can conclude that there exist $i_1, i_2 \in [dR]$ such that $I_{i_1}^{\leq dk}(f) \geq \frac{\delta}{d}$ and $I_{i_2}^{\leq dk}(g) \geq \frac{\delta}{d}$ such that $(i_1, i_2) \in d \leftrightarrow d$. Let $\sigma_1, \sigma_2 \in [dR]$ be such that $i_1 = \pi_1(\sigma_1), i_2 \in \sigma_2$. As $f(x^{\pi_1}) = g_u(x)$, $I_{\pi_1^{-1}(i_1)}^{\leq dk}(g_u) \geq \frac{\delta}{d}$. And thus, $\sigma_1 \in L(u)$, and similarly $\sigma_2 \in L(v)$. As $(i_1, i_2) \in d \leftrightarrow d$, $(\sigma_1, \sigma_2) \in \Psi_e$, which completes the proof.

4.4 Reducing multigraph (exact) d-to-1 to $(d+1)$-to-1 conjecture

For the version of d-to-1 conjecture where we only require the constraint maps to be at most d-to-1, the d-to-1 conjecture trivially implies the $(d+1)$-to-1 conjecture. O'Donnell and Wu [OW09] remark that no such reduction appears to be known for the exact d-to-1 conjecture. Here we prove that the exact d-to-1 conjecture implies the exact $(d+1)$-to-1 conjecture when the underlying Label Cover instances are allowed to have parallel edges. We remark that multigraph version of exact d-to-1 conjecture, which is implied by the simple graph version, also suffices for our reduction to graph coloring (and indeed all known reductions from d-to-1 Label Cover).

Let $G = ((V = X \cup Y, E), (dR, R), \Psi)$ be a Label Cover instance such that every constraint is of d-to-1 structure. We reduce it to $G' = ((V = X \cup Y, E'), ((d+1)R, R), \Psi')$ such that

1. If G is satisfiable, G' is satisfiable as well.
2. If every labeling violates at least ϵ fraction of constraints in G, then every labeling violates at least $\epsilon' = 2\epsilon$ fraction of constraints in G'.

Reduction. We first change the label set of X from $[dR]$ to $[(d+1)R]$. For every constraint ψ in G between nodes $u \in X$ and $v \in Y$, we replace it with R constraints $\psi_1, \psi_2, \ldots, \psi_R$ between u and v in the following way: the relation between old labels is the same as ψ i.e. when $x \leq dR$, $(x, y) \in \psi_j$ for $j = 1, 2, \ldots, R$ if and only if $(x, y) \in \psi$. When $x > dR$, $(x, y) \in \psi_j$ if and only if R divides $(x + j - y)$. This ensures that each new label is mapped to a different label in each of the R new constraints. The constraints are clearly of $(d+1) - to - 1$ form.

Completeness. If there is a labeling satisfying all the constraints of G, the same labeling satisfies all the constraints in G' as well.

Soundness. Suppose that there is no labeling satisfying at least ϵ fraction of constraints in G. Note that this implies that R is at least $\frac{1}{\epsilon}$ as there is always a labeling satisfying at least $\frac{1}{R}$ fraction of constraints: fix a labeling to the vertices on the left, and assign a label to the vertices in R uniformly at random from $[R]$. We claim that there is no labeling satisfying more than 2ϵ fraction of constraints in G'. Consider an arbitrary labeling of G, $\sigma : V \to [(d+1)R]$. We can divide the set of edges E' of G' into two parts: the edges (u, v) such that $\sigma(u) \leq dR$ and the edges (u, v) such that $\sigma(u) > dR$. Let the set of first type of edges where the left vertex is assigned the new label be denoted by E_1, and the set of second type of edges be denoted by E_2. In E_1, the fraction of constraints that can be satisfied by σ is at most $\frac{1}{R} \leq \epsilon$. Note that we can get a labeling σ' of G by replacing labels of vertices in X with label greater than dR with an arbitrary label in $[dR]$, and keeping rest of the labels intact. For the edges in E_2, the labelings σ and σ' coincide. As σ' can satisfy at most ϵ fraction of constraints of G, σ can only satisfy at most ϵ fraction of overall edges in E'. Thus, overall σ satisfies at most $\epsilon + \frac{1}{R} \leq 2\epsilon$ fraction of constraints in E', which proves the required soundness claim.

Chapter 5

Rainbow coloring hardness via low sensitivity polymorphisms

5.1 Introduction

Similar to the Graph coloring problem, hypergraph coloring also received a lot of attention in Graph Theory and Theoretical Computer Science. Even though there is a simple algorithm to check if a given graph is 2-colorable or not, checking if a 3-uniform hypergraph can be colored with two colors so that no hyperedge is monochromatic is one of the classic NP-hard problems. This raises the question of identifying if 2-coloring is easy on special hypergraphs of interest. For example, if a k-uniform hypergraph is k-partite, i.e., the vertices can be partitioned into k parts so that every hyperedge intersects each part, then there are simple algorithms to properly color the hypergraph with two colors. Suppose we know that a k-uniform hypergraph is promised to be $k-1$-partite, can we color it with two colors?

An equivalent way to formulate this question is in terms of rainbow coloring. A k-uniform hypergraph is said to be *r-rainbow colorable* if there is a coloring of vertices with r colors such that all the r colors appear in every edge. Unlike usual coloring, rainbow coloring becomes harder as we have more colors. Note that r-partiteness is the same thing as r-rainbow colorability. As mentioned above, a k-uniform hypergraph that is promised to be k-rainbow colorable can be efficiently colored with two colors. One big hammer approach for this is to use semidefinite programming and find a unit vector for each vertex such that sum of the vectors in each edge sum to zero, and then use random hyperplane rounding. But the 2-coloring can also be performed by a simple random walk algorithm — start with an arbitrary coloring, and as long as there is a monochromatic edge, pick an arbitrary one and flip the color of a random vertex in it. This process will converge to a 2-coloring in a quadratic number of iterations with high probability [McD93].

If we relax the k-rainbow colorability assumption slightly to that of $(k-1)$-rainbow colorability, there are no known efficient algorithms for 2-coloring. It is tempting to conjecture that in fact this task is hard (in fact, even if we are allowed c colors for any constant c; this was shown assuming the V Label Cover conjecture in [BG17]). If we relax the rainbow colorability assumption further, Austrin, Bhangale and Potukuchi proved that it is NP-hard to 2-color a k-uniform hypergraph when it is promised to be $(k - 2\sqrt{k})$-rainbow colorable [ABP20]. They also showed that it is NP-hard to 2-color a 4-uniform hypergraph even if it is 3-rainbow colorable. In this work, we focus on hardness results for the $(k-1)$-rainbow colorable case, as this promise is the closest to k-partiteness which makes 2-coloring easy. While we can't show hardness of 2-coloring, we show that rainbow coloring with $\lceil \frac{k-2}{2} \rceil$ colors is hard. Formally, our main result is the following.

Theorem 35. *Fix an integer $k \geq 4$. Given a k-uniform hypergraph that is promised to be $(k-1)$-rainbow colorable, it is NP-hard to rainbow color it with $\lceil \frac{k-2}{2} \rceil$ colors.*

As a corollary, we also get the following, which extends the similar result of [ABP20] for the $k = 4$ case (their techniques did not generalize beyond the 4-uniform case).

Theorem 36. *For $k \leq 6$, given a k-uniform hypergraph that is promised to be $k-1$-rainbow colorable, it is NP hard to 2-color it.*

5.1.1 Techniques

There have been broadly three lines of attack on proving hardness for graph and hypergraph coloring problems.

The first line of work gives reductions from Label Cover analyzed using Fourier-analytic techniques of the sort originally pioneered by Håstad [Hås01]. Early applications of this method showed strong hardness results for coloring 2-colorable hypergraphs of low uniformity with any constant number of colors [GHS02; Hol02; Kho02a; Sak14]. This approach, augmented with the invariance principle of [MOO10] and some of its extensions such as [DMR09; Mos10; Wen13], was used to prove further hardness results for hypergraph coloring [BK10; GL18] and strong conditional hardness results for graph coloring [DMR09]. These methods usually also prove a stronger statement about finding independent sets in the graphs or hypergraphs. For rainbow coloring, it is proved in [GL18] by combining together many of these techniques that a $k/2$-rainbow colorable k-uniform hypergraph cannot be colored with any constant number of colors in polynomial time unless P = NP. While our results in Chapter 4 used the latest PCSP ideas, the core technique used, however, is this analytical approach.

A less extensive line of work proceeds via combinatorial gadgets that are analyzed using ideas based on the chromatic number of Kneser graphs and similar results. The first exemplar of this approach was the hardness of $O(1)$-coloring 2-colorable 3-uniform hypergraphs shown in [DRS05]. Unlike the analytic results for 4-uniform hypergraphs mentioned above, this result does not show hardnes of finding large independent sets. (This was later shown in [KS14] using the analytic approach, albeit under the d-to-1 conjecture.) A few recent results have revived this combinatorial approach, by re-deriving and improving some previous hardness results for hypergraph coloring using simpler proofs [Bha18; ABP19].

The third and most recent line of work adapts the universal algebraic method behind the complexity classification of constraint satisfaction problems that culminated in the resolution of the Feder-Vardi CSP dichotomy conjecture [Bul17; Zhu20]. Here, the coloring problem is viewed as a Promise Constraint Satisfaction Problem (PCSP), and its associated polymorphisms are then analyzed. [1] In the cases when the polymorphisms are severely limited, one can show hardness via a reduction from Label Cover. The approach to study PCSP using polymorphisms originated in [AGH17], and was used to show hardness results for graph and hypergraph coloring in [BG16]. The algebraic theory was further developed significantly in [Bar+21] leading among other results to a proof of NP-hardness of 5-coloring 3-colorable graphs.

In this work, we follow the algebraic approach to prove Theorem 35. Note that rainbow coloring is a natural Promise Constraint Satisfaction Problem. As proved in [Bar+21; BG21a], as with normal CSP, the complexity of a promise CSP is captured by its associated polymorphisms. Recall that polymorphisms of a PCSP are ways to combine multiple solutions of an instance satisfying the stronger predicate to obtain a solution to the instance satisfying the weaker predicate. The high level principle behind the algebraic approach is that the problem should be easy when it has a rich enough set of polymorphisms that include functions with strong symmetries, and hard when all its polymorphisms are somehow skewed and lack symmetries. This has been fully established for CSPs — when there are polymorphisms which obey weak near-unanimity, the CSP is polytime solvable, and otherwise NP-complete.[2] The hardness part of this dichotomy is easier and was known for a while; the much harder algorithmic part was established only recently in [Bul17; Zhu20].

For promise CSPs, which form a much richer class, our current understanding is rather limited, for both the algorithmic and hardness sides. It is not clear (to even conjecture) what kind of lack in symmetries in the polymorphisms might dictate hardness, and how one might show the corresponding hardness. A simple (but rather limited) sufficient condition for hardness is when all the polymorphisms are dictators that depend on a single coordinate. In [AGH17], it has been proved that if all the polymorphisms of a PCSP are juntas[3], then the PCSP is NP-hard. This is the basis of the hardness results for $(2+\epsilon)$-SAT [AGH17] and 3-coloring graphs that admit a homomorphism to C_k for any fixed odd integer k [KO19]. The recent hardness of 5-coloring 3-colorable graphs in [Bar+21] proceeds by showing that the absence of arity 6 polymorphisms with the so-called Olšák symmetry implies NP-hardness, and then verifying that 3 vs. 5-coloring lacks such polymorphisms.

It turns out that the polymorphisms of rainbow coloring can have Olšák symmetries *and* be non-juntas. We will get around this by proving that these polymorphisms are C-fixing in the sense that there exists a constant number of coordinates and an assignment to them such that if we fix these coordinates to the assignment, the value of the function is fixed. This is also studied as certificate complexity in Boolean function analysis [APV16]. We then prove that if the polymorphisms of a PCSP are C-fixing, then the PCSP is NP-hard.

In order to prove that the polymorphisms have low certificate complexity, we use the connection between sensitivity and certificate complexity of functions. These two ways of characterizing

[1]The proof in [DMR09] also implicitly studies polymorphisms and proves that they must have a small number of coordinates with sizeable influence and thus are not too symmetric. This influence-type characterization interfaces better with Unique Games or other highly structured forms of Label Cover.

[2]For the case of Boolean CSPs, the CSP is hard iff all polymorphisms are unary, i.e., either the dictator function or its complement.

[3]A C-junta is a function that depends on at most C inputs

complexity of functions are well studied in the context of Boolean functions. It is worth emphasizing that for our purposes, all we need is to show that low sensitivity (even sensitivity 2 suffices) implies constant certificate complexity, and thus we are not interested in optimal gaps between sensitivity and certificate complexity. The famous sensitivity vs. block sensitivity conjecture [Nis89] states that these two parameters are in fact polynomially related. In one of the earliest works related to this problem, Simon [Sim83] proved that certificate complexity is at most exponential in sensitivity. We extend this to larger domains, and then use it to prove that the polymorphisms that we study have low certificate complexity.

The second step is to then use the C-fixing property to show NP-hardness of the PCSP. This is done by the usual paradigm of reducing from Label Cover using polymorphism tests (better known as long code tests) of functions associated with vertices of the Label Cover instance. A more structured form of the C-fixing property, where the C variables are fixed to the same value, is used in [BG21a] to show NP-hardness of certain Boolean PCSPs. However, in order to prove NP-hardness using our more general notion of C-fixing, we end up needing stronger properties of the Label Cover instance. As a result, our reduction is from the *smooth* Label Cover problem that was introduced and shown to be NP-hard in [Kho02a], and has found many applications in inapproximability since.

A natural question is to understand how far we can push these techniques. Our NP hardness reduction from smooth Label Cover works when the polymorphisms of the PCSP in hand are C-fixing for some constant C. As k increases, the polymorphisms of PCSP of 2-coloring a k-uniform hypergraph that is promised to be $(k-1)$-rainbow colorable get richer. When k is at most 6, the polymorphisms are C-fixing. At $k = 7$, we show that there is a polymorphism that is not C-fixing for any constant C. In fact, one would need C to be linear in the arity of polymorphisms which also rules out using smooth Label Cover with very strong soundness.

5.1.2 Prior work on rainbow coloring and related problems

Various notions of approximate coloring with rainbow colorability guarantees have been studied in the literature. Bansal and Khot [BK10] prove that when the input hypergraph is promised to be almost k-rainbow colorable, it is Unique Games hard to color it with $O(1)$ colors. Sachdeva and Saket [SS13] establish NP-hardness of $O(1)$ coloring a k-uniform hypergraph when it is promised to be almost $\frac{k}{2}$-rainbow colorable. This was extended by Guruswami and Lee [GL18] to perfectly $\frac{k}{2}$-rainbow colorable hypergraphs. Guruswami and Saket [GS17] prove similar results assuming stronger forms of rainbow colorability in the completeness case. In [ABP20], Austrin, Bhangale and Potukuchi proved that it is NP-hard to 2-color a k-uniform hypergraph when it is promised to be $(k-2\sqrt{k})$ rainbow colorable. On the other hand, when the hypergraph is promised to be $k-\sqrt{k}$ rainbow colorable, Bhattiprolu, Guruswami and Lee [BGL15] give algorithms to color the hypergraph with two colors that miscolors at most $k^{-\Omega(k)}$ fraction of edges; this beats the 2^{k+1} fraction achieved by random coloring that is the best possible for general 2-colorable hypergraphs [Hås01]. Brakensiek and Guruswami [BG17] put forth a problem called V label cover (to possibly serve as a perfect completeness variant surrogate for Unique games), and under its conjectured inapproximability proved that it is hard to color a k-uniform $(k-1)$-rainbow colorable hypergraph with $O(1)$ colors.

A related notion of hypergraph coloring is strong coloring where we color a k-uniform hypergraph with $s > k$ colors such that in any edge, all the k vertices are colored with distinct colors. Brakensiek and Guruswami [BG16] prove that it is NP-hard to 2-color a k-uniform hypergraph that is promised to be strongly colorable with $\lceil \frac{3k}{2} \rceil$ colors. Assuming the V Label Cover conjecture, it is hard to $O(1)$-color k-uniform hypergraphs with strong chromatic number at most $k + \sqrt{k}$ [BG17].

5.1.3 Outline

We start with a few notations and definitions in Section 5.2. In Section 5.3, we study polymorphisms of rainbow coloring. We first prove a result on sensitivity and certificate complexity and use it to prove properties of polymorphisms of the PCSP that we are studying. Then, we use these in Section 5.4 to prove NP hardness. Finally, we use the rainbow coloring of hypergraphs ideas to show hardness for Vector Bin Covering problem.

5.2 Preliminaries

5.2.1 Rainbow Coloring PCSP

In $\mathsf{RAINBOW}(k, r, q)$ problem, the input is a k uniform hypergraph. The goal is to distinguish between the cases when the hypergraph is rainbow colorable with r colors and when it cannot be rainbow colorable with q colors. More formally, we can define the problem as below:

Definition 37. *($\mathsf{RAINBOW}(k, r, q)$) In the $\mathsf{RAINBOW}(k, r, q)$ promise CSP, $q \le r \le k$, we have a pair of predicates (A, B) defined as follows:*

- $A : [r]^k \to \{0, 1\} : A(x_1, x_2, \cdots, x_k) = 1$ *if and only if* $\{x_1, x_2, \cdots, x_k\} = [r]$.
- $B : [q]^k \to \{0, 1\} : B(y_1, y_2, \cdots, y_k) = 1$ *if and only if* $\{y_1, y_2, \cdots, y_k\} = [q]$.

Note that we need q, r to be at most k since we cannot rainbow color a k uniform hypergraph with more than k colors. We also need the condition that $q \le r$ for the promise problem to make sense: If the hypergraph is r rainbow colorable, we can infer that it is already $q < r$ rainbow colorable too. Thus, the promise problem is to identify if the hypergraph is r rainbow colorable or it *cannot even be* rainbow colorable with q colors. Furthermore, in this work we will be only dealing with near perfect completeness case when hypergraph is $(k-1)$-partite i.e. $r = k - 1$.

We now direct our attention to polymorphisms of $\mathsf{RAINBOW}(k, r, q)$. By definition, the polymorphisms of hypergraph coloring PCSPs turn out to be colorings of certain tensor product hypergraphs. Fix n to be arity of the polymorphisms. We can infer that the polymorphisms of $\mathsf{RAINBOW}(k, r, q)$ are proper q-rainbow colorings of the following k uniform hypergraph $\mathtt{RH}_n(k, r)$:

- The vertex set of hypergraph is the set $V = [r]^n$.

- A k element set $(\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_k)$, where each $\mathbf{v}_i \in [r]^n$ is an edge if and only if for every $j \in [n]$, the set $\{(\mathbf{v}_1)_j, (\mathbf{v}_2)_j, \cdots, (\mathbf{v}_k)_j\}$ is equal to $[r]$.

That is, a set of k vectors from $[r]^n$ forms an edge if in the matrix obtained by plugging these vectors as rows, all the r elements from $[r]$ occur in every column.

5.2.2 Complexity measures of functions

Finally, we define the notions of sensitivity and C-fixing of functions.

Definition 38. *(Sensitivity at $\boldsymbol{x}$) For a function $f : [r]^n \to [q]$, and an input $\boldsymbol{x} \in [r]^n$, the sensitivity of f at $\boldsymbol{x}$, denoted by $S(f, \boldsymbol{x})$ is defined as the number of coordinates i such that changing $\boldsymbol{x}$ at i can change the value of f i.e. $S(f, \boldsymbol{x}) = \big|\{i \in [n] | \exists a : f(\boldsymbol{x}) \neq f(\boldsymbol{x} : x_i \leftarrow a)\}\big|$.*

Definition 39. *(Sensitivity) The sensitivity of a function $f : [r]^n \to [q]$, denoted by $S(f)$ is defined as the maximum sensitivity of f over all $\boldsymbol{x}$ in $[r]^n$ i.e. $S(f) = \max_{\boldsymbol{x}} S(f, \boldsymbol{x})$.*

Definition 40. *(C fixing) A function f from $[r]^n$ to $[q]$ is said to be C-fixing for some integer C if there exists a set $S = \{s_1, s_2, \cdots, s_C\} \subseteq [n]$ and a vector $\alpha = \{\alpha_1, \alpha_2, \cdots, \alpha_C\} \subseteq [r]^m$ such that $f(\boldsymbol{x}) = p$ whenever $x_{s_i} = \alpha_i$ for all integers $1 \leq i \leq t$, for some fixed $p \in [q]$.*

5.3 Polymorphisms

In this section, we will analyze the properties of polymorphisms of rainbow coloring. In order to do so, we will prove prove that low sensitivity implies low certificate complexity. Using this, we will establish that the polymorphisms for $\mathsf{RAINBOW}(k, k-1, \lceil \frac{k-2}{2})\rceil$ are C-fixing. Along the way, we will study rainbow colorings of various hypergraphs related to $\mathtt{RH}_n(k, r)$. Finally, we will show that our techniques cannot prove hardness of $\mathsf{RAINBOW}(7, 6, 2)$ by presenting a polymorphism that is not C-fixing for any constant C.

5.3.1 Sensitivity vs certificate complexity

We extend a lemma of [Sim83] that proves that if a function has low sensitivity then the function is C fixing, to larger domains. The proof is along the same lines as the original proof.

Lemma 41. *Let $f : [r]^n \to [q]$ be a function with sensitivity s, and let $b \in [q]$ such that $f^{-1}(b)$ is non empty. Then, $|f^{-1}(b)| \geq r^{n-s}$.*

Proof. Fix s, and induct on n. The case $n = s$ is trivial. Let $\mathbf{x} \in [r]^n$ be such that $f(\mathbf{x}) = b$. Since $s < n$, there is a coordinate in $\mathbf{x}$ that is not sensitive. Without loss of generality, let it be 1, and let $\mathbf{x} = (x_1, \mathbf{y})$. As the first coordinate is not sensitive for $\mathbf{x}$, we can conclude that $f(\alpha, \mathbf{y}) = b$ for all $\alpha \in [r]$.

Consider the set of functions $g_i : [r]^{n-1} \to [q], g_i(\mathbf{u}) = f(i, \mathbf{u}), i \in [r]$. Note that for each such g_i, the set $g_i^{-1}(b)$ is non-empty. In addition, for every $i \in [r]$, sensitivity of g_i is at most the sensitivity of f. Thus, by induction, we know that each such g_i has at least r^{n-1-s} elements $\mathbf{u}$ in

$[r]^{n-1}$ such that $g_i(\mathbf{u}) = b$. Note that every such $\mathbf{u}$ gives $f(i, \mathbf{u}) = b$. By combining over all is, we can conclude that there are at least $r \cdot r^{n-1-s} = r^{n-s}$ elements $\mathbf{x} \in [r]^n$ such that $f(\mathbf{x}) = b$. $\square$

Lemma 42. *Let $f : [r]^n \to [q]$ be a function with sensitivity $s < n/2$. Then, it is C-fixing for $C = s(r-1)r^{2s+1}$.*

Proof. We will actually prove a stronger statement that f is a C-junta. Let the set A denote the set of coordinates with non-zero influence of f i.e. the coordinates that are sensitive for some input. Our goal is to upper bound the cardinality of A.

For a function $f : [r]^n \to [q]$, let the set of sensitive edges $E(f)$ be defined as the set of pairs of elements $\mathbf{x}, \mathbf{y} \in [r]^n$ such that $f(\mathbf{x}) \neq f(\mathbf{y})$, and $\mathbf{x}, \mathbf{y}$ differ on exactly one coordinate. From the sensitivity bound on f, we can deduce that

$$|E(f)| \leq s(r-1)r^n \tag{5.1}$$

Fix an arbitrary coordinate $i \in A$. There are elements $\mathbf{x}, \mathbf{y} \in [r]^n$ such that $x_i = \alpha, y_i = \beta, \alpha \neq \beta, f(\mathbf{x}) \neq f(\mathbf{y})$, and $\mathbf{x}, \mathbf{y}$ differ only in ith coordinate. Define a function $g : [r]^{n-1} \to \{0, 1\}$ as $g(\mathbf{z})$ is 1 if and only if $f(\alpha, \mathbf{z}) = f(\mathbf{x})$, and $f(\beta, \mathbf{z}) = f(\mathbf{y})$ where we use the notation $(\alpha, \mathbf{z})$ to denote the vector in $[r]^n$ obtained by inserting α in ith position to $\mathbf{z} \in [r]^{n-1}$. Now, since $f(\alpha, \mathbf{z})$ and $f(\beta, \mathbf{z})$ are both sensitive to at most s coordinates, $g(\mathbf{z})$ is sensitive to at most $2s$ coordinates. Also note that $g^{-1}(1)$ is non-empty. Thus, by Lemma 41, we can conclude that $|g^{-1}(1)|$ is at least r^{n-1-2s}. In other words, each sensitive coordinate contributes at least r^{n-2s-1} edges to $E(f)$. Thus, we can conclude that

$$|E(f)| \geq |A| r^{n-2s-1} \tag{5.2}$$

Combining Equation (5.1) and Equation (5.2), we get

$$|A| \leq s(r-1)r^{2s+1} \tag{5.3}$$

which proves the required claim. $\square$

5.3.2 Low sensitivity polymorphisms of rainbow coloring

We now turn our attention towards our main goal in this section: to show that polymorphisms of $\mathsf{RAINBOW}(k, k-1, q)$ are C-fixing for $q = \lceil \frac{k-2}{2} \rceil$. As we have already mentioned earlier, the polymorphisms of rainbow coloring themselves are rainbow colorings of certain tensor product hypergraphs. To be precise, the n-ary polymorphisms of $\mathsf{RAINBOW}(k, r, q)$ are precisely q-rainbow colorings of $\mathtt{RH}_n(k, r)$. Thus our new goal is to prove that any integer $q \geq 2$, any q-rainbow coloring of $\mathtt{RH}_n(2q+2, 2q+1)$ is a C-fixing function.

In order to achieve this, we will first define certain hypergraphs similar to $\mathtt{RH}_n(k, r)$.

Definition 43. *$\mathtt{H}_n(r, s) = (V, E)$ is a r uniform hypergraph where the vertex set V is equal to $[r]^n$. A set of vectors $(\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_r)$ is an edge if and only if*

1. *In every coordinate $i \subseteq [n]$, at least $r-1$ elements occur i.e. $\left|\bigcup_j (\boldsymbol{u}_j)_i\right| \geq r-1 \quad \forall i \in [n]$.*

2. *All the r elements occur in at least $n-s$ coordinates i.e. $\left|\bigcup_j (\boldsymbol{u}_j)_i\right| = r$ for at least $n-s$ choices of i in $[n]$.*

The reason behind studying these hypergraphs is that the q-rainbow colorings of $\mathtt{RH}_n(2q+2, 2q+1)$ are very closely related to q-rainbow colorings of $\mathtt{H}_n(2q+1, c)$ for any absolute constant c. In fact if we can prove that q-rainbow colorings of $\mathtt{H}_n(2q+1, c)$ are C-fixing, it implies that q-rainbow colorings of $\mathtt{RH}_n(2q+2, 2q+1)$ are $\max(C, c)$-fixing. This is formally proved in Lemma 47. Thus our modified objective is to argue that q-rainbow colorings of $\mathtt{H}_n(2q+1, c)$ are C-fixing. In order to do so, we inductively relate q-rainbow colorings of $\mathtt{H}_n(t, c)$ and $\mathtt{H}_n(t-1, c-1)$. As a base case, we have the following lemma:

Lemma 44. *For all integers $q \geq 2$ and $n \geq 1$, the hypergraph $\mathtt{H}_n(2q-1, 1)$ cannot be rainbow colored with q colors.*

Proof. We will use induction on q. For the case $q = 2$, rainbow coloring with 2 colors is the same as proper coloring the hypergraph with 2 colors. The fact that $\mathtt{H}_n(3, 1)$ cannot be two colored follows from [ABP20] (Lemma 3.2 with $d = 3$).

Suppose for contradiction that f is a valid q-rainbow coloring of $\mathtt{H}_n(2q-1, 1)$. Let $r = 2q-1$ denote the uniformity of the hypergraphs. Consider the set of r vectors in $[r]^n : \{\bigcup_i (i, i, \cdots, i)\}$. As there are at most $q < r$ colors, some two elements of this set should have same f value. Without loss of generality, let $f(r-1, r-1, \cdots, r-1) = f(r, r, \cdots, r) = c$ for some $c \in [q]$. Consider the $r-2$-uniform hypergraph $H = \mathtt{H}_n(r-2, 1)$. Note that every edge in H together with $\mathbf{u} = (r-1, r-1, \cdots, r-1)$ and $\mathbf{v} = (r, r, \cdots, r)$ forms an edge in $\mathtt{H}_n(r, 1)$. Thus, all the $q-1$ colors in $[q] \setminus \{c\}$ occur in every edge of coloring of $\mathtt{H}_n(r-2, 1)$ using f. This implies that we can get a a valid $(q-1)$-rainbow coloring of $\mathtt{H}_n(r-2 = 2(q-1)-1, 1)$ by restricting f to $[r-2]^n$, and replacing the color c using arbitrary color from $[q] \setminus \{c\}$. However, by induction hypothesis such a coloring cannot exist, and thus we have arrived at contradiction. $\square$

Now, we will use this to argue about q-rainbow colorings of $\mathtt{H}_n(2q+1, 3)$ via q-rainbow colorings of $\mathtt{H}_n(2q, 2)$. Consider the hypergraph $\mathtt{H}_n(2q, 2)$. A trivial way to q-rainbow color this hypergraph is to pick a coordinate $i \in [n]$, and partition the set $[2q]$ into q disjoint sets of size two, let's say $A_1, A_2, \cdots, A_q$ and assign the value $p \in [q]$ to $f(\mathbf{x})$ if and only if $x_i \in A_p$. It turns out that this is the only way to q-rainbow color $\mathtt{H}_n(2q, 2)$. We prove it in the lemma below:

Lemma 45. *Let f be a q-rainbow coloring of $\mathtt{H}_n(r = 2q, 2)$. Then, there exists an index $i \in [n]$, sets $A_1, A_2, \cdots, A_q \subseteq [r]$ each of size 2, and mutually disjoint such that $f(\mathbf{x}) = j$ iff $x_i \in A_j$.*

Proof. First we will prove that the sensitivity of f is at most 1. Let $\mathbf{x} = (x_1, x_2, \cdots, x_n)$ be an arbitrary vector in $[r]^n$. Consider a $r-1$-uniform hypergraph $H(\mathbf{x})$ defined on $([r] \setminus x_1) \times ([r] \setminus x_2) \times \cdots \times ([r] \setminus x_n)$. We add a $r-1$ vector set as edge of $H(\mathbf{x})$ if and only if it has at most one coordinate where there are missing elements i.e. all the $[r] \setminus x_i$ occur in all but one coordinate i, and in that coordinate, at most one value is missing.

Note that $H(\mathbf{x})$ is isomorphic to $\mathtt{H}_n(2q-1, 1)$. From Lemma 44, we know that $H(\mathbf{x})$ cannot be rainbow colored with q colors. Thus, when we view f as a coloring of $H(\mathbf{x})$, there is an edge that has a color missing. Let it be denoted by $e = (\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_{r-1})$. Let j be the coordinate where there is a missing element in e. If there is no coordinate with missing element, j can be arbitrary. Without loss of generality, let color $1 \subseteq [q]$ be missing in e. Note that $\{\mathbf{x}\} \cup e$ is an edge of $\mathtt{H}_n(r,1)$, and thus an edge of $\mathtt{H}_n(r,2)$ as well. Since f is a proper q-rainbow coloring of $\mathtt{H}_n(2q,2)$, we can conclude that $f(\mathbf{x}) = 1$. In fact, we can actually deduce something stronger. Let $\mathbf{y} \in [r]^n$ such that $\mathbf{x}$ and $\mathbf{y}$ differ on exactly one coordinate $j' \in [r]^n, j' \neq j$. Note that $\{\mathbf{y}\} \cup e$ is also a valid edge of $\mathtt{H}_n(2q,2)$ since it has at most two coordinates where there are missing elements i.e. j' and j. Thus, $f(\mathbf{y}) = 1 = f(\mathbf{x})$. Thus, for every $\mathbf{x}$, in except for one coordinate, changing the value of the coordinate preserves the value of $\mathbf{x}$. In other words, the sensitivity of f is at most 1.

Using this, we will now prove that f is a dictator. Let i be an influential coordinate of f i.e. there exists $\mathbf{x}, \mathbf{y} \in [r]^n$ differing only in ith coordinate such that $f(\mathbf{x}) \neq f(\mathbf{y})$. We claim that $f(\mathbf{u}) = f(\mathbf{x})$ for all $\mathbf{u} \in [r]^n$ such that $u_i = x_i$, and $f(\mathbf{u}) = f(\mathbf{y})$ if $u_i = y_i$. We will prove this by induction on the number of coordinates in which $\mathbf{x}$ and $\mathbf{u}$ differ excluding i. Since f has sensitivity at most 1, the only sensitive coordinate of $\mathbf{x}$ and $\mathbf{y}$ is i. Thus, for any $\mathbf{u}$ differing only in one coordinate from $\mathbf{x}$ (other than i) such that $u_i = x_i$ or y_i will have corresponding f value. Suppose that the statement holds for all $\mathbf{u}$ differing from $\mathbf{x}$ in t coordinates excluding i.

Now, let $\mathbf{u}$ differ from $\mathbf{x}$ in $t+1$ coordinates excluding i. We can find $\mathbf{v} \in [r]^n, \mathbf{w} \in [r]^n$ such that $\mathbf{v}$ and $\mathbf{x}$ differ in t coordinates excluding i, $v_i = x_i$; $\mathbf{w}$ and $\mathbf{y}$ differ in t coordinates excluding i, $w_i = y_i$, and one of $\mathbf{v}$ and $\mathbf{w}$ differs from $\mathbf{u}$ in at most one coordinate. By induction hypothesis, $f(\mathbf{v}) = f(\mathbf{x}), f(\mathbf{w}) = f(\mathbf{y})$. Since $\mathbf{v}$ and $\mathbf{w}$ differ in a single coordinate i, i is the only sensitive coordinate of $\mathbf{v}$ and $\mathbf{w}$. Thus, $f(\mathbf{u})$ is equal to either $f(\mathbf{v})$ or $f(\mathbf{w})$ depending on $u_i = x_i$ or y_i. This completes the inductive proof.

To complete the proof that f is a dictator, we will use this to show that there cannot be two influential coordinates. Suppose that there are two influential coordinates i and j. From previous argument, we can infer that there are assignments $i_1, i_2, j_1, j_2 \in [r]$ such that assigning these to corresponding coordinates fixes the value of f. Also note that assigning i as i_1 and i_2 fixes f to different values. Similarly, assigning j as j_1 and j_2 fixes f to different values. This gives rise to contradiction since if we set coordinate i to i_1, f should be fixed irrespective of j is equal to j_1 or j_2. Thus, there can be only one influential coordinate for f, or in other words f is a dictator.

Let p be the dictator coordinate of f i.e. there exists a function $g : [r] \to [q]$ such that $f(\mathbf{x}) = g(x_p)$. From the definition of the hypergraph $\mathtt{H}_n(r,2)$, for every $j \in [r]$, the set $\{\bigcup_i g(x_i)\} \setminus g(x_j)$ should be equal to $[q]$. This proves that there exists sets $A_1, A_2, \cdots, A_q \subseteq [r]$ each of size two, and mutually disjoint such that $g(\alpha) = j$ if and only if $\alpha \in A_j$, which proves the required claim. $\square$

We finish the chain of inductive arguments by proving a key property of q-rainbow colorings of $\mathtt{H}_n(2q+1,3)$.

Lemma 46. *Let $f : [2q+1]^n \to [q]$ be a q-rainbow coloring of $\mathtt{H}_n(r = 2q+1, 3)$. Then, there exists an index $i \in [n]$, and $\alpha \in [r]$ such that $S(f, \mathbf{x}) \leq 2$ for all $\mathbf{x} \in [r]^n$ such that $x_i = \alpha$.*

Proof. Let $\mathbf{x} = (x_1, x_2, \cdots, x_n) \in [r]^n$ be an arbitrary vector in $[r]^n$. Similar to the previous lemma, we define the complement hypergraph associated with $\mathbf{x}$. Consider a $r-1$-uniform hypergraph $H(\mathbf{x})$ defined on $([r] \setminus x_1) \times ([r] \setminus x_2) \times \cdots \times ([r] \setminus x_n)$. We add a $r-1$ vector set as edge of $H(\mathbf{x})$ if and only if it has at most two coordinates where there are missing elements i.e. all the $[r] \setminus x_i$ occur in all but two coordinates i, and in these two coordinates, at least $r-2$ values occur. Note that $H(\mathbf{x})$ is isomorphic to $\mathtt{H}_n(r-1, 2)$.

We can view $f : [2q+1]^n \to [q]$ as a q-rainbow coloring of $H(\mathbf{x})$. If f is not a valid q-rainbow coloring of $H(\mathbf{x})$, by the same argument as in Lemma 45, we can deduce that $S(f, \mathbf{x}) \leq 2$. If f is a valid coloring of $H(\mathbf{x})$, we will use the properties proved in Lemma 45. Let us define a function $g : [r]^n \to [n] \cup \{\perp\}$ such that for a vector $\mathbf{x} \in [r]^n$,

1. If f is a valid q-rainbow coloring of $H(\mathbf{x})$, then Lemma 45 implies that there exists a coordinate $i \in [n]$ such that f is a dictator in ith coordinate in $H(\mathbf{x})$. In this case, set $g(\mathbf{x}) = i$.
2. If f is not a valid q-rainbow coloring of $H(\mathbf{x})$, let $g(\mathbf{x}) = \perp$.

First, we will prove that there exists an index $i \in [n]$ such that $g(\mathbf{x}) \in \{i, \perp\}$ for all $\mathbf{x} \in [r]^n$. Suppose $g(\mathbf{x}) = i \in [n]$, and $g(\mathbf{y}) = j \in [n]$ where $\mathbf{x}, \mathbf{y} \in [r]^n$ and $i \neq j$. Since $g(\mathbf{x}) = i$, there exist sets $S_1, S_2, \cdots, S_n \subseteq [r]$ such that f is a dictator on ith coordinate in $S = S_1 \times S_2 \times \cdots \times S_n \subseteq [r]^n$. In particular, there is a subset $A \subseteq S_i$ such that $|A| = 2$, and $f(\mathbf{x})$, $\mathbf{x} \in S$, is equal to 1 if and only if $x_i \in A$. Similarly, there exist sets $T_1, T_2, \cdots, T_n \subseteq [r]$ such that f is a dictator on jth coordinate in $T = T_1 \times T_2 \times \cdots \times T_n \subseteq [r]^n$. There exists a subset $B \subseteq T_j$ such that $|B| = 2$, and $f(\mathbf{x}), \mathbf{x} \in T$ is equal to $c \neq 1$ if and only if $x_j \in B$ for some $c \in [q]$. Combining the both, let $U_i = S_i \cap T_i$, $|U_i| \geq r - 2 \, \forall \, i \in [n]$. We can deduce that f is a dictator in both i and j coordinates in $U = U_1 \times U_2 \times \cdots \times U_n$. This implies that f is a constant function in U. Recall that there are two assignments in S_i that make f equal to 1 and two assignments in T_j that make f equal to $c \neq 1$. Thus, $f(\mathbf{x}')$ is equal to 1 for some $\mathbf{x}' \in U$ and $f(\mathbf{y}') = c \neq 1$ for some $\mathbf{y}' \in U$. This contradicts the fact that f is a constant function in U. Thus, there exists an index $i \in [n]$ such that $g(\mathbf{x})$ is either equal to i or is equal to $\perp$ for all $\mathbf{x} \in [r]^n$. Without loss of generality let that be the first coordinate i.e. for all $\mathbf{x} \in [r]^n$, $g(\mathbf{x}) \in \{1, \perp\}$.

Consider the case when $g(\mathbf{x}) = \perp$ for every $\mathbf{x} \in [r]^n$. In this case, we know that $S(f, \mathbf{x}) \leq 2$ for all $\mathbf{x} \in [r]^n$. In particular, we can set α arbitrary and say that $S(f, \mathbf{x}) \leq 2$ whenever $x_1 = \alpha$. So we are only left with the case when there exists a $\mathbf{x} \in [r]^n$ such that $g(\mathbf{x}) = 1$. We will now prove that there exists $\alpha \in [r]$ such that $g(\mathbf{x}) = \perp$ whenever $x_1 = \alpha$, thus proving the required sensitivity bound.

Suppose for contradiction that for every $\alpha \in [r]$, there exists $\mathbf{x} \in [r]^n$ such that $x_1 = \alpha$, and $g(\mathbf{x}) = 1$. Consider a pair $\mathbf{u}, \mathbf{v} \in [r]^n$ such that $u_1 = \alpha, v_1 = \beta \neq \alpha$ and $g(\mathbf{u}) = g(\mathbf{v}) = 1$. Let $\mathbf{u} = (u_1, u_2, \cdots, u_n)$, $S_i = [r] \setminus u_i$ and f is dictator on 1st coordinate in $S = S_1 \times S_2 \times \cdots \times S_n$. There is a function $h_1 : S_1 \to [q]$ such that $f(\mathbf{x}) = h_1(x_1)$ if $\mathbf{x} \in S$ and $|h_1^{-1}(c)| = 2 \, \forall c \in [q]$. Similarly, let $\mathbf{v} = (v_1, v_2, \cdots, v_n)$, $T_i = [r] \setminus v_i$ and f is dictator on first coordinate in $T = T_1 \times T_2 \times \cdots \times T_n$. There is a function $h_2 : T_1 \to [q]$ such that $f(\mathbf{x}) = h_2(x_1)$ if $\mathbf{x} \in T$ and $|h_2^{-1}(c)| = 2 \quad \forall c \in [q]$. Let $U_i = S_i \cap T_i$. Note that $U = U_1 \times U_2 \times \cdots U_n$ is non empty and f is dictator on 1st coordinate in U as well. Note that $|U_i| \geq r - 2$ for all $i \in [n]$. Thus, we can conclude that if $\gamma \in U_1$, then $h_1(\gamma) = h_2(\gamma)$.

Applying this to all pairs $\mathbf{u}, \mathbf{v}$ such that $g(\mathbf{u}) = g(\mathbf{v}) = 1$, we can infer that there exists a function $h : [r] \to [q]$ that satisfies the property that for all $\mathbf{x} \in [r]^n$ such that $g(\mathbf{x}) = 1$, let $\mathbf{x} = (x_1, x_2, \cdots, x_n)$, $S_i = [r] \setminus x_i$, $S = S_1 \times S_2 \times \cdots \times S_n$, then $f(\mathbf{y}) = h(y_1)$ for all $\mathbf{y} \in S$. As $r = 2q + 1 > 2q$, there exists $b \in [q]$ such that $|h^{-1}(b)| \geq 3$. Let $\gamma \in [r]$ be such that $h(\gamma) \neq b$. From our assumption that for every $\alpha \in [r]$ there exists $\mathbf{x} \in [r]^n$ such that $g(\mathbf{x}) = 1$ and $x_1 = \alpha$, there exists $\mathbf{u} \in [r]^n$ such that $u_1 = \gamma$ and $g(\mathbf{u}) = 1$. Now, let $\mathbf{u} = (u_1, u_2, \cdots, u_n)$, $S_i = [r] \setminus u_i$, $S = S_1 \times S_2 \times \cdots \times S_n$, and we know that $f(\mathbf{x}) = h(x_1)$ if $\mathbf{x} \in S$, and $\left|h^{-1}(c) \cap S_1\right| = 2\, \forall c \in [q]$. However, this contradicts the fact that $h(u_1) = h(\gamma) \neq b$, and $|h^{-1}(b)| = 3$. Thus, there exists $\alpha \in [r]$ such that $g(\mathbf{x}) = \perp$ for all $\mathbf{x} \in [r]^n$ such that $x_1 = \alpha$. $\square$

Finally, we will use the previous hypergraph coloring properties to argue about polymorphisms of rainbow coloring.

Lemma 47. *Suppose $f : [2q+1]^n \to [q]$ is an n-ary polymorphism of* ***RAINBOW***$(2q+2, 2q+1, q)$ *i.e. f is a proper q-rainbow coloring of $\mathtt{RH}_n(2q+2, 2q+1)$. Then, there exist constant $C = C(q)$ independent of n such that f is C-fixing.*

Proof. Let $r = 2q + 1$. Let $f : [r]^n \to [q]$ be a polymorphism of RAINBOW$(2q + 2, 2q + 1, q)$. We can view f as a q-rainbow coloring of $\mathtt{H}_n(r, 3)$ as the vertex set of $\mathtt{RH}_n(r + 1, r)$ and of $\mathtt{H}_n(r, 3)$ is equal to $[r]^n$. If it is not a valid q-rainbow coloring, there is an edge in which not all q colors appear. Let that edge be $e = (\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_r\}$ and $c \in [q]$ be a missing color in $\{f(\mathbf{v}_1), f(\mathbf{v}_2), \cdots, f(\mathbf{v}_r)\}$. Since this edge is part of $H_n(r, 3)$, except for 3 values of i, for all other i, the set $\big((\mathbf{v}_1)_i, (\mathbf{v}_2)_i, \cdots, (\mathbf{v}_r)_i\big)$ is equal to $[r]$. Let the missing coordinates be the set $S = \{i_1, i_2, i_3\}$. Now consider an element $\mathbf{u}$ of $[r]^n$ such that it has missing values of e in S. From the definition of $\mathtt{RH}_n(r + 1, r)$, we can deduce that the set $e \cup \mathbf{u}$ is an edge of $\mathtt{RH}_n(r + 1, r)$. Since f is a valid q-rainbow coloring of $\mathtt{RH}_n(r + 1, r)$, $f(\mathbf{u})$ is equal to c. Note that this should hold irrespective of what values $\mathbf{u}$ has, in coordinates outside S. This proves that f is C-fixing with $C = 3$.

On the other hand if f is a valid q-rainbow coloring of $\mathtt{H}_n(r, 3)$, using Lemma 46, we can deduce that there exists an index $i \in [n]$, and $\alpha \in [r]$ such that $S(f, \mathbf{x}) \leq 2$ whenever $x_i = \alpha$. Now, we can consider a function $g : [r]^{n-1} \to [q]$ which on an input $\mathbf{y} \in [r]^{n-1}$, is equal to $f(\mathbf{x}), \mathbf{x} = \mathbf{y}, x_i \leftarrow \alpha \in [r]^n$ i.e. we first insert α in ith position to $\mathbf{y}$ and then apply f. Note that g has sensitivity at most 2. From Lemma 42, we can conclude that g is C-fixing for $C = 2(r-1) \cdot r^5$. In other words, g is fixed by assigning values to a set of C indices. This implies that f is also $C' = C + 1$-fixing since we can first assign ith index to α, then use C-fixing property of g. $\square$

5.3.3 High sensitivity polymorphism of RAINBOW$(7, 6, 2)$

We show that there exists a function $f : [6]^n \to \{0, 1\}$ that is a polymorphism of RAINBOW$(7, 6, 2)$ that is not C-fixing for any constant C. We start with a dictator but add just enough noise such that the function still remains being a polymorphism, but it is no longer C-fixing. Let $wt(\mathbf{x})$ denote the number of $i \in [n], i > 1$ such that $x_i = 1$. Let $S \subseteq [6]^n$ denote the set of

$\mathbf{x} \in [6]^n$ such that $wt(\mathbf{x}) > \frac{2n}{3}$. Let $h : [6]^n \to \{0, 1\}$ be noise function defined below. For a given $\mathbf{x} \in [6]^n$, we define $f(\mathbf{x})$ as follows:

1. If $\mathbf{x} \notin S$
   (a) If $x_1 \leq 3$, $f(\mathbf{x}) = 0$
   (b) Else, $f(\mathbf{x}) = 1$
2. Else $f(\mathbf{x}) = h(\mathbf{x})$.

A choice of noise function that works is inverting the original function: $h(\mathbf{x})$ is defined as 1 if and only if $x_1 \leq 3$.

Proposition 48. *The function $f : [6]^n \to \{0, 1\}$ defined above is a polymorphism of $\mathsf{RAINBOW}(7, 6, 2)$ and it is not C-fixing for any $C < \frac{n}{3}$.*

Proof. Any polymorphism of $\mathsf{RAINBOW}(7, 6, 2)$ is a proper 2-rainbow coloring of $\mathtt{RH}_n(7, 6)$. Recall that rainbow coloring with two colors is the same as standard hypergraph coloring with two colors.

Polymorphism. In any set of 7 vectors E in $[6]^n$ such that all the 6 elements occur in all the coordinates, at most two vectors can be in S. This is because, in any set of three vectors in S, there exists a coordinate in which all three values are equal to 1. Thus, there are vectors $\mathbf{x} \notin S$ with $x_1 \leq 3$ and vector $\mathbf{y} \notin S$ such that $y_1 \geq 3$ in E, which together ensures that E is not monochromatic.

C-fixing. Suppose that there exists a set $T = \{t_1, t_2, \cdots, t_m\} \subseteq [n]$ and $(\alpha_1, \alpha_2, \cdots, \alpha_m) \subseteq [6]^m$ such that $f(\mathbf{x}) = b$ for all $\mathbf{x}$ such that $x_i = \alpha_i$ for all $1 \leq i \leq m$, for some fixed $b \in \{0, 1\}$. We will prove that $|T| \geq \frac{n}{3}$. Suppose for contradiciton that $|T| < \frac{n}{3}$. First, if $1 \notin T$, we can set all coordinates outside T to be equal to $\beta \neq 1$, and in this case $f(\mathbf{x}) = x_1$, which cannot be fixed if $1 \notin T$. Thus $1 \in T$. Next, if all the coordinates outside T are all equal to 1, then $f(\mathbf{x})$ is equal to noise function, which is different from the case when the rest are equal to $\beta \neq 1$. Thus, if f is indeed a C-fixing function, for the C-fixing assignment, the value of f should be independent of the assignment to the coordinates outside T. However, that is not the case as the value of f changes when we set all the coordinates outside T to be 1 or $\beta \neq 1$. $\square$

5.4 NP-Hardness

In this section, we will use the properties of polymorphisms proved so far to argue about NP hardness of rainbow coloring PCSP. We will prove the below theorem:

Theorem 49. *Suppose that there exists a constant C such that for all integers $n \geq 1$, every n-ary polymorphism of $\mathsf{RAINBOW}(k, k-1, q)$ is C-fixing. Then, the corresponding decision problem $\mathsf{RAINBOW}(k, k-1, q)$ is NP hard.*

Before delving into the proof of Theorem 49, we first mention that this theorem together with Lemma 47 implies Theorem 35. In Lemma 47, we have proved that for every $q \geq 2$, the polymorphisms of $\mathsf{RAINBOW}(2q+2, 2q+1, q)$ are C-fixing. This fact combined with Theorem 49 implies that $\mathsf{RAINBOW}(2q+2, 2q+1, q)$ is NP hard for every $q \geq 2$. This already proves Theorem 35 when k is even. In order to prove when k is odd, note that we can use Lemma 45 in Lemma 47 to prove that the polymorphisms of $\mathsf{RAINBOW}(2q+1, 2q, q)$ are C-fixing. We can combine this with Theorem 49 to prove Theorem 35 when k is odd.

The rest of this section is dedicated to proving Theorem 49. Like various other hardness of approximation results, we will use the standard label cover with long code framework. We reduce *smooth* label cover introduced in [Kho02a] to rainbow coloring PCSP.

First we formally define the gap Label Cover problem below:

Definition 50. *($(1, \epsilon_{LC})$ Gap Label Cover) In $(1, \epsilon_{LC})$ Gap Label Cover, we are given a Label Cover instance $(G = (L, R, E), \Sigma, \Pi)$, and the goal is to distinguish between the following two cases:*

1. *There is a labelling $\sigma : G \to \Sigma$ that satisfies all the constraints.*
2. *No labelling can satisfy ϵ_{LC} fraction of constraints.*

As mentioned earlier, we need stronger properties of the Label Cover instance that we are starting with. One such property is smoothness.

Definition 51. *(Smoothness) A Label Cover instance $(G = (L, R, E), \Sigma, \Pi)$ is said to be $(J, \epsilon)-$smooth if for any vertex $u \in L$ and a set of labels $S \subseteq \Sigma, |S| \leq J$, over a uniformly random neighbor $v \in R$, $Pr(\left|\bigcup_{s\in S} \Pi_{u,v}(s)\right| < |S|) \leq \epsilon$.*

The following is an easier version of Theorem 1.17 in [Wen13].

Theorem 52. *For every $\epsilon, \epsilon_{LC} > 0$ and $J \in \mathbb{Z}_+$, there exists $n = n(\epsilon, \epsilon_{LC}, J)$ such that $(1, \epsilon_{LC})$ Gap Label Cover with Label size n that is promised to be (J, ϵ)-smooth is NP hard.*

Reduction. We start with $(1, \epsilon_{LC})$ Gap Label Cover instance $(G = (L, R, E), \Sigma, \Pi)$ that is promised to be (C, ϵ)-smooth, for ϵ and ϵ_{LC} to be set later, and output a PCSP instance. Let n denote the label size $n = |\Sigma|$. For each vertex $v \in L \cup R$, we add a set of nodes K_v of size $[k-1]^n$, indexed by n length vectors. We add two types of constraints:

1. Coloring constraints: Inside every vertex of the Label Cover instance, we add the following constraints among the $[k-1]^n$ nodes. We add the constraint that the promise relation should be satisfied in the set T of k nodes $\{\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_k\}$ in $[k-1]^n$, if for every $i \in [n]$, the set $\{\bigcup_j (\mathbf{u}_j)_i\}$ has cardinality $k-1$.
2. Equality constraints: For every constraint $\Pi_e : u \to v$ of the Label Cover, we add a set of equality constraints between nodes $\mathbf{u} \in K_u, \mathbf{v} \in K_v$ if for all $i \in [n]$, $u_i = v_{\Pi_e(i)}$.

Note that the Coloring constraints give rise to rainbow colorings of k uniform hypergraphs. It is yet unclear how we can justify adding equality constraints. One way to handle the equality constraints is to have a single node for all the vertices corresponding to equality constraint. However, this fails if we want to add a coloring constraint that involves two copies of the same vertex. A neater way to get around this is to argue that adding equality constraints does not change the set of polymorphisms, and thus the hardness of the predicate remains same with or without equality constraints. This simple fact is proved in Section 5.6.

Completeness. If the label cover instance is satisfiable, then PCSP instance that is being output can be satisfied by assignment from $[k-1]$. Suppose $\sigma : L \cup R \to \Sigma$ is a labelling that satisfies all the constraints of the Label Cover. For every vertex $\mathbf{u} \in K_u$ corresponding to the vertex $u \in L \cup R$, we can assign the value $u_{\sigma(u)}$. In other words, in every long code, we are assigning corresponding dictator function. The coloring constraints are defined precisely such that this assignment satisfies the constraints. The equality constraints also follows since the labelling σ satisfies all the constraints of the Label Cover.

Soundness. If the Label Cover is not ϵ_{LC} satisfiable, we need to show that there is no assignment of the PCSP instance in $[q]$ that satisfies all the constraints. Taking contrapositive, if there is an assignment in $[q]$ to PCSP instance that satisfies all the constraints, then we will prove that there is an assignment to the Label Cover instance that can satisfy c fraction of constraints, for an absolute constant c. Taking $\epsilon_{LC} < c$, we can arrive at a contradiction, thus proving that there is no assignment in $[q]$ to PCSP that satisfies all the constraints.

Let $f_v : [k-1]^n \to [q]$ denote the assignment to the PCSP instance that satisfies all the constraints for $v \in L \cup R$. From the coloring constraints, we can infer that f_v is a n-ary polymorphism of $\mathsf{RAINBOW}(k, k-1, q)$. Thus, it is C-fixing for a constant C independent of n.

For every vertex $v \in L \cup R$ of the Label Cover instance, we will assign a set of labels $A(v)$. For vertices v in L, $A(v)$ is the C-fixing set. Since the Label Cover instance is smooth, we will only consider the constraints where all these labels go to distinct labels on the right under projections. We can set the smoothness parameter ϵ to be 0.1 for example, and we will be left with $\frac{9}{10}$ fraction of original constraints. We will prove that there is an assignment that satisfies c fraction of these constraints, for an absolute constant c, which will prove the original soundness claim. Thus in all the remaining constraints, the set of labels in $A(v)$ go to distinct labels on the right. Thus, for a vertex $v \in R$, each constraint (u, v) gives rise to C coordinates $\Pi_{u,v}(A(u))$. For these vertices, we set $A(v)$ to be the set of maximal disjoint sets of such a projections of C coordinates.

In order to prove that there is a good labelling to the Label Cover, we assign a label to every vertex v from $A(v)$ uniformly at random and prove that it satisfies constant fraction of constraints with non-zero probability. We will in fact show that the random assignment satisfies a constant fraction of constraints in expectation. We prove this in two steps. First, we show that for every constraint (u, v) of the Label Cover, there exists $x \in A(u), y \in A(v)$ such that $\Pi_{u,v}(x) = y$. This follows from the definitions of $A(v)$: suppose the projection of $A(u)$ is disjoint from $A(v)$. In that case, we can add the projection of $A(u)$ to $A(v)$ to get a larger set in v, which contradicts the fact that $A(v)$ is the maximal such union of disjoint projections. This implies that the uniformly random labelling satisfies each constraint (u, v) of Label Cover with probability at least $\frac{1}{|A(u)||A(v)|}$.

To complete the proof, we need to bound the sizes of $A(v)$. As we have already mentioned, for $v \in L, |A(v)| \leq C$. We bound the size of $A(v)$ for vertices v in R using the below lemma.

Lemma 53. *Suppose $f : [k-1]^n \to q$ is a polymorphism of $\mathsf{RAINBOW}(k, k-1, q)$. Let $A_1, A_2, \cdots, A_t$ be mutually disjoint subsets of $[n]$ such that each of them is a C-fixing set of f. Then, $t < k$.*

Proof. First note that all the A_is should fix f to the same value in $[q]$ since otherwise, the vector $\mathbf{u} \in [k-1]^n$ that has all the fixing sets in A_is is forced to be equal to multiple colors in $[q]$ at the same time. Let all the A_is be C-fixing with respect to value $b \in [q]$ i.e. for each $i \in [t]$, there exists an assignment to A_i such that if the value of $\mathbf{x}$ in A_i is equal to the assignment, then the value of $f(\mathbf{x})$ is equal to b irrespective of values of coordinates outside A_i. If $t \geq k$, we can find $\mathbf{u}_1, \mathbf{u}_2, \cdots \mathbf{u}_k$ such that all $[k-1]$ occur in every coordinate, and $\mathbf{u}_i$ has the fixing assignment of A_i. This implies that $f(\mathbf{u}_i) = b$ for all i. However, note that $\{\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_k\}$ is an edge of $\mathtt{RH}_n(k, k-1)$, and thus if f is a polymorphism of $\mathsf{RAINBOW}(k, k-1, q)$, all the $[q]$ elements should occur in $\{f(\mathbf{u}_1), f(\mathbf{u}_2), \cdots, f(\mathbf{u}_k)\}$. This is a contradiction since for all i, $f(\mathbf{u}_i) = b$. □

From the lemma, we can infer that the cardinality of $A(v)$ for $v \in R$ is at most kC. Combining this with the above, we can deduce that there is an assignment that satisfies $\frac{1}{kC^2}$ fraction of constraints, which is a constant fraction of constraints, independent of n.

5.5 Application: Vector Bin Covering

5.5.1 Problem overview

In the Bin Covering problem, the input is a set of n items with size $a_1, a_2, \ldots, a_n$. The objective is to partition them into a maximum number of parts such that in each part, the sum of the items is at least 1. The problem is a classic NP-Hard problem and admits a Polynomial Time Approximation Scheme(PTAS) [CJK01]. Vector Bin Covering is a multidimensional variant of Bin Covering. In this problem, the input is a set of n vectors in $[0,1]^d$. The objective is to partition these into the maximum number of parts such that in each part, the sum of vectors is at least 1 in every coordinate.

Definition 54. *(Vector Bin Covering) In the Vector Bin Covering problem, we are given n vectors $v_1, v_2, \ldots, v_n \in [0,1]^d$. The goal is to partition the input vectors into maximum number of parts $A_1, A_2, \ldots, A_m$ such that*

$$\sum_{j \in A_i} v_j \geq \boldsymbol{1}^d \ \forall i \in [m]$$

The problem is also referred to as "dual Vector Packing" in the literature. It is introduced by Alon *et al*. [Alo+98] who gave a $O(\log d)$ factor approximation algorithm. They also gave a d factor algorithm using a method from the area of compact vector approximation that outperforms the above algorithm for small values of d. On the hardness front, Ray [Ray21] showed that the 2-dimensional Vector Bin Covering problem is hard to approximate within a factor of $\frac{998}{997}$.

We show $\Omega\left(\frac{\log d}{\log\log d}\right)$ hardness, almost matching the $O(\log d)$ factor algorithm [Alo+98].

Theorem 55. *d-dimensional Vector Bin Covering is NP-hard to approximate within $\Omega\left(\frac{\log d}{\log\log d}\right)$ factor.*

5.5.2 Hardness of Vector Bin Covering via Rainbow Coloring

Our hard instances for Vector Bin Covering are when the vectors are from $\{0,1\}^d$. In this setting, the Vector Bin Covering problem is closely related to the hypergraph rainbow coloring problem.

We now give a simple reduction from approximate rainbow coloring to Vector Bin Covering.

Lemma 56. *Given a hypergraph $H = (V, E)$ and a parameter k, there is a polynomial time reduction that outputs a Vector Bin Covering instance $v_1, v_2, \ldots, v_n \in \{0,1\}^d$ with $n = |V|, d = |E|$ such that*

1. *(Completeness.) If H is k-rainbow colorable, there is a partition of $[n]$ into k parts $A_1, A_2, \ldots, A_k$ such that*
$$\sum_{j \in A_i} v_j \geq \boldsymbol{1}^d \ \forall i \in [k]$$
2. *(Soundness.) If H is not 2-colorable, there is no partition of $[n]$ into A_1, A_2 such that*
$$\sum_{j \in A_i} v_j \geq \boldsymbol{1}^d \ \forall i \in [2]$$

Proof. Let $n = |V|, d = |E|$. We order the edges E as $E = \{e_1, e_2, \ldots, e_d\}$. We output a set of vectors $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ where each $v_i \in \{0,1\}^d$ is defined as follows:

$$(v_i)_j = \begin{cases} 1 & \text{if } i \in e_j \\ 0 & \text{otherwise.} \end{cases}$$

We analyze this reduction:

1. (Completeness.) Suppose that the hypergraph H has a rainbow coloring with k colors $f : V \to [k]$. We partition $[n]$ into k parts $A_1, A_2, \ldots, A_k$ such that
$$A_i = \{j \in [n] : f(j) = i\}$$
Consider an arbitrary integer $i \in [k]$. Note that for every edge e in H, $e \cap A_i \neq \phi$. Thus,
$$\sum_{j \in A_i} v_j \geq \mathbf{1}^d$$
2. (Soundness.) Suppose that the hypergraph H has no proper coloring with 2 colors. Then, we claim that there is no partition of $[n]$ into two parts A_1, A_2 such that
$$\sum_{j \in A_i} v_j \geq \mathbf{1}^d \ \forall i \in [2]$$
Suppose for contradiction that there exists A_1, A_2 with the above property. Consider the coloring of the hypergraph $f : V \to [2]$ as
$$f(v) = \begin{cases} 1 & \text{if } v \in A_1 \\ 2 & \text{if } v \in A_2 \end{cases}$$

Consider an arbitrary edge $e_l, l \in [d]$ of the hypergraph H. As $\sum_{j \in A_i} (v_j)_l \geq 1$ for all $i \in [2]$, there exist $v_1, v_2 \in e_l$ such that $v_1 \in A_1, v_2 \in A_2$. Thus, the coloring f is a proper 2 coloring of the hypergraph H, a contradiction. $\square$

We combine this reduction with the hardness of approximate rainbow coloring to prove the hardness of Vector Bin Covering, namely Theorem 55. Note that the dimension of the resulting vectors in the Vector Bin Covering instance $\mathcal{V}$ is equal to the number of edges $m = |E|$ of the hypergraph H, and the gap in the optimal Bin Covering value of $\mathcal{V}$ is equal to k, the number of colors. Hence, to obtain better inapproximability results for Vector Bin Covering that grow with d, our goal is to show the hardness of approximate rainbow coloring on hypergraphs with m edges where the number of colors k is as large a function of m as possible. Towards this, we prove that it is NP-hard to 2-color a hypergraph with m edges that is promised to be rainbow colorable with $k = \Omega\left(\frac{\log m}{\log \log m}\right)$ colors.

Theorem 57. *Given a hypergraph H with m edges, it is NP-hard to distinguish between the following:*

1. *(Completeness) H is k-rainbow colorable.*
2. *(Soundness) H is not 2-colorable.*

where $k = \Omega\left(\frac{\log m}{\log \log m}\right)$.

We defer the proof of Theorem 57 to Section 5.5.3.

We now prove the hardness of Vector Bin Covering using Theorem 57.

Proof of Theorem 55. Using Theorem 57 combined with the reduction in Lemma 56, we get that the following problem is NP-hard. Given a set of n vectors $v_1, v_2, \ldots, v_n \in \{0,1\}^d$, distinguish between

1. $\mathcal{V}$ can be partitioned into $k = \Omega\left(\frac{\log d}{\log \log d}\right)$ parts such that in each part, the sum of vectors is at least 1 in every coordinate.
2. $\mathcal{V}$ cannot be partitioned into 2 parts such that in each part, the sum of vectors is at least 1 in every coordinate. In other words, the maximum number of parts into which $\mathcal{V}$ can be partitioned such that in each part, the sum of vectors is at least 1 in every coordinate is equal to 1.

Thus, it is NP-hard to approximate d-dimensional Vector Bin Covering within $k = \Omega\left(\frac{\log d}{\log \log d}\right)$. $\square$

5.5.3 Proof of Theorem 57

Our proof follows the same lines as that of Theorem 49. We present the full proof here for the sake of completeness.

We first need a slightly different notion of C-fixing for $C = 1$.

Definition 58. *(1-fixing [BG16; GS20b]) A function $f : [k]^n \to \{0,1\}$ is said to be 1-fixing if there exists an index $\ell \in [n]$ and values $\alpha, \beta \in [k]$ such that*

$$f(\boldsymbol{x}) = 0\,\forall \boldsymbol{x} : \boldsymbol{x}_\ell = \alpha \quad \textit{and} \quad f(\boldsymbol{x}) = 1\,\forall \boldsymbol{x} : \boldsymbol{x}_\ell = \beta$$

In the analysis of our reduction later, we need a definition and a lemma from [ABP20].

Definition 59. *(The hypergraph $H_r^n[k]$) The hypergraph $H_r^n[k] = (V, E)$ is a k-uniform hypergraph with vertex set as the set of n-dimensional vectors over $[k]$ i.e. $V = [k]^n$. A set of k vectors $\boldsymbol{v}^1, \boldsymbol{v}^2, \ldots, \boldsymbol{v}^k$ form an edge of the hypergraph if*

$$\sum_{i=1}^{n} \left|[k] \setminus \{\boldsymbol{v}_i^j : j \in [k]\}\right| \leq r$$

Lemma 60. *For every $k \geq 2$, the hypergraph $H^n_{\lfloor \frac{k}{2} \rfloor}[k]$ is not 2-colorable.*

We analyze the polymorphisms of the underlying PCSP used in our reduction.

Lemma 61. *Fix $k \geq 3$. Suppose $f : [k]^n \to \{0,1\}$ satisfies the below two-coloring property: For every $2k$ vectors $\boldsymbol{v}^1, \boldsymbol{v}^2, \ldots, \boldsymbol{v}^{2k} \in [k]^n$ with*

$$\{\boldsymbol{v}_i^j : j \in [2k]\} = [k]\,\forall i \in [n],$$

we have

$$\{f(\boldsymbol{v}^j) : j \in [2k]\} = \{0,1\}.$$

Then, f is 1-fixing.

Proof. We first prove that there exist $\ell \in [n], \alpha \in [k]$, $b \in \{0,1\}$ such that $f(\mathbf{x}) = b$ for all $\mathbf{x} \in [k]^n$ with $\mathbf{x}_\ell = \alpha$. Suppose for contradiction that this is not the case. Then, for every $i \in [n], j \in [k]$ there exist vectors $\mathbf{x}^{i,j}, \mathbf{y}^{i,j} \in [k]^n$ such that $\mathbf{x}_i^{i,j} = \mathbf{y}_i^{i,j} = j$, and $f(\mathbf{x}^{i,j}) = 0$ where as $f(\mathbf{y}^{i,j}) = 1$.

Let $r = \lfloor \frac{k}{2} \rfloor$. We view $f : [k]^n \to \{0,1\}$ as an assignment of two colors to the vertices of the hypergraph $H_r^n[k]$. As the hypergraph is not two colorable (Lemma 60), we can infer that there is an edge of $H_r^n[k]$ all of whose vertices are assigned the same color. In other words, there exist k vectors $\mathbf{v}^1, \mathbf{v}^2, \ldots, \mathbf{v}^k \in [k]^n$ and $b \in \{0,1\}$ such that $f(\mathbf{v}^j) = b$ for all $j \in [k]$. Furthermore, there are at most r missing values in these vectors i.e.

$$\sum_{i=1}^{n} \left|[k] \setminus \{\mathbf{v}_i^j : j \in [k]\}\right| \leq r$$

Now, we pick r vectors $\mathbf{u}^1, \mathbf{u}^2, \ldots, \mathbf{u}^r$ (with repetitions if needed) by filling the missing values using $\mathbf{x}^{i,j}, \mathbf{y}^{i,j}$ vectors such that

1. $f(\mathbf{u}^j) = b$ for all $j \in [r]$.
2. For every $i \in [n]$,
$$\{\mathbf{v}_i^j : j \in [k]\} \cup \{\mathbf{u}_i^j : j \in [r]\} = [k]$$

By taking the union of $\{\mathbf{v}^1, \mathbf{v}^2, \ldots, \mathbf{v}^k\}$ and $\{\mathbf{u}^1, \mathbf{u}^2, \ldots, \mathbf{u}^r\}$, and repeating some vectors, we obtain $2k$ vectors $\mathbf{w}^1, \mathbf{w}^2, \ldots, \mathbf{w}^{2k}$ with $f(\mathbf{w}^j) = b$ for all $j \in [2k]$, and

$$\{\mathbf{w}_i^j : j \in [2k]\} = [k] \ \forall i \in [n]$$

However, this contradicts the two-coloring property of f. Thus, there exist $\ell \in [n], \alpha \in [k], b \in \{0, 1\}$ such that $f(\mathbf{x}) = b$ for all $\mathbf{x} \in [k]^n$ with $\mathbf{x}_\ell = \alpha$.

We now claim that there exists $\beta \in [k]$ such that $f(\mathbf{x}) = 1 - b$ for all $\mathbf{x} \in [k]^n$ with $\mathbf{x}_\ell = \beta$. Suppose for contradiction that this is not the case. Then, there exist k vectors $\mathbf{v}^1, \mathbf{v}^2, \ldots, \mathbf{v}^k$ such that $\mathbf{v}_\ell^j = j$ for all $j \in [k]$, and $f(\mathbf{v}^j) = b$ for all $j \in [k]$. We now pick $\mathbf{v}^{k+1}, \mathbf{v}^{k+2}, \ldots, \mathbf{v}^{2k} \in [k]^n$ such that $\mathbf{v}_\ell^j = \alpha$ for all $j \in \{k+1, k+2, \ldots, 2k\}$, and $\mathbf{v}_i^j = j - k$ for all $i \in [n]$ with $i \neq \ell$, and $j \in \{k+1, k+2, \ldots, 2k\}$. These $2k$ vectors $\mathbf{v}^1, \mathbf{v}^2, \ldots, \mathbf{v}^{2k}$ satisfy

1. $f(\mathbf{v}^j) = b$ for all $j \in [2k]$.
2. For every $i \in [n]$,
$$\{\mathbf{v}_i^j : j \in [2k]\} = [k]$$

contradicting the two-coloring property of f. Thus, there exists $\beta \in [k]$ such that $f(\mathbf{x}) = 1 - b$ for all $\mathbf{x} \in [k]^n$ with $\mathbf{x}_\ell = \beta$, completing the proof that f is 1-fixing. □

We are now ready to prove Theorem 57. Our hardness result is obtained using a reduction from the Label Cover problem, similar to Theorem 49.

Reduction. We start with the Label Cover instance $G = (V = L \cup R), E, \Sigma = \Sigma_L = \Sigma_R, \Pi)$ from Theorem 128 and output a hypergraph $H = (V', E')$. Let n denote the label size $n = |\Sigma|$. For each vertex $v \in L \cup R$, we have a long code containing a set of nodes K_v of size $[k]^n$, indexed by n length vectors.

1. The vertex set of the hypergraph V' is the union of all the long code nodes.
$$V' = \bigcup_{v \in V} K_v$$
2. Edges of the hypergraph: For every vertex $v \in V$ of the Label Cover instance, we add an edge in E' for each set of $2k$ vectors $\{\mathbf{v}^1, \mathbf{v}^2, \ldots, \mathbf{v}^{2k}\}$ in K_v, if
$$\{\mathbf{v}_i^j : j \in [2k]\} = [k] \, \forall i \in [n]. \tag{5.4}$$
The number of edges in H is at most
$$|E'| \leq |V| \binom{k^{|\Sigma|}}{2k} \leq |V| k^{O(k)}$$
3. Equality constraints: For every constraint $\Pi_e : u \to v$ of the Label Cover, we add a set of equality constraints between nodes $\mathbf{x} \in K_u$, $\mathbf{y} \in K_v$ if for all $i \in [n]$, $\mathbf{x}_i = \mathbf{y}_{\Pi_e(i)}$. By adding an equality constraint between two nodes, we identify the two nodes together and treat it as a single node. That is, we compute the connected components of the equality

constraints graph and identify a single master node for each component. We then obtain a multi-hypergraph H_1 from H by replacing each node with the corresponding master node. However, a vertex could appear multiple times in an edge in H_1. We delete such occurrences from H_1 by setting each edge to be a simple set of the vertices contained in it, and obtain the final hypergraph H_2. We note the following:

(a) There exists a k-rainbow coloring of H, $f : V' \to [k]$ that respects the equality constraints i.e. $f(\mathbf{x}) = f(\mathbf{y})$ for all pairs of nodes $\mathbf{x}, \mathbf{y}$ with equality constraints between them if and only if H_2 is k-rainbow colorable.

(b) Similarly, there exists a 2-coloring of H that respects equality constraints if and only if H_2 is 2-colorable.

Finally, the number of edges in H_2 is at most the number of edges in H.

Completeness. Suppose that there is a labeling $\sigma : V \to \Sigma$ that satisfies all the constraints. We define the coloring $f : V' \to [k]$ of H as follows. For every node $\mathbf{x} \in K_v$, we set the dictatorship function

$$f(\mathbf{x}) = \mathbf{x}_{\sigma(v)}$$

By the constraints added in Equation (5.4), the function f is a valid k-rainbow coloring of H. As σ satisfies all the constraints of the Label Cover, the coloring f satisfies all the equality constraints.

Soundness. Suppose that there is no labeling $\sigma : V \to \Sigma$ that satisfies all the constraints in G. Then we claim that there is no 2-coloring of H that respects all the equality constraints. Suppose for contradiction that there is a 2-coloring $f : V' \to \{0, 1\}$ that respects all the equality constraints.

Consider a vertex $v \in V$. The function $f_v : [k]^n \to \{0, 1\}$, defined as f on K_v satisfies the conditions in Lemma 61. Thus, f_v is 1-fixing for every $v \in V$. Hence, there is a function $L : V \to \Sigma$ such that for every $v \in V$, f_v is 1-fixing on the coordinate $L(v)$. We now claim that the labeling $\sigma : V \to \Sigma$ defined as $\sigma(v) = L(v)$ satisfies all the constraints in G.

Consider an edge $e = (u, v)$, $u \in L, v \in R$ with the projection constraint $\Pi_e : \Sigma \to \Sigma$. Our goal is to show that $\Pi_e(L(u)) = L(v)$. Suppose for contradiction that $\Pi_e(L(u)) \neq L(v)$. By the 1-fixing property of f_u, we have $\alpha_u, \beta_u \in [k]$ such that

$$f_u(\mathbf{x}) = 0\, \forall \mathbf{x} \in [k]^n : \mathbf{x}_{L(u)} = \alpha_u \quad \text{and} \quad f_u(\mathbf{x}) = 1\, \forall \mathbf{x} \in [k]^n : \mathbf{x}_{L(u)} = \beta_u$$

Similarly, we have $\alpha_v, \beta_v \in [k]$ such that

$$f_v(\mathbf{y}) = 0\, \forall \mathbf{y} \in [k]^n : \mathbf{y}_{L(v)} = \alpha_v \quad \text{and} \quad f_v(\mathbf{y}) = 1\, \forall \mathbf{y} \in [k]^n : \mathbf{y}_{L(v)} = \beta_v$$

By the equality constraints, $f_u(\mathbf{x}) = f_v(\mathbf{y})$ for all $\mathbf{x}, \mathbf{y} \in [k]^n$ such that $\mathbf{x}_i = \mathbf{y}_{\Pi_e(i)}\, \forall i \in [n]$. Let $\mathbf{y}' \in [k]^n$ be an arbitrary vector with $\mathbf{y}'_{\Pi_e(L(u))} = \alpha_u, \mathbf{y}'_{L(v)} = \beta_v$. We choose $\mathbf{x}' \in [k]^n$ such that for all $i \in [n]$, $\mathbf{x}'_i = \mathbf{y}'_{\Pi_e(i)}$. Note that $\mathbf{x}'_{L(u)} = \alpha_u$. Thus, $f_u(\mathbf{x}') = 0$ where as $f_v(\mathbf{y}') = 1$. However, this contradicts the equality constraints.

5.6 Adding equality constraints

We will prove that adding equality relation structure does not affect the polymorphisms. By equality relation structure, we mean $(=,=) := (A \subseteq [q_1]^2, B \subseteq [q_2]^2)$ where $q_1 \leq q_2$ and $A = \{(x,x) : x \in [q_1]\}, B = \{(y,y) : y \in [q_2]\}$.

Lemma 62. *Suppose $P = (A_1, B_1), (A_2, B_2), \cdots, (A_m, B_m)$ is a PCSP template. Let the template Q be obtained by adding relational structure $(A', B') := (=,=)$ to P. Then, under log space reductions, P is equivalent to Q.*

Proof. We will show that P and Q have identical set of polymorphisms. Note that as Q contains all the relations structures in P, polymorphisms of Q are a subset of P. We claim that the reverse direction also holds because every function is a polymorphism of $(=,=)$. Consider a polymorphism $f : [q_1]^n \to [q_2]$ be an n-ary polymorphism of P. Consider n vectors $\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_n$ such that for all $i \in [n]$, $\big((\mathbf{v}_i)_1, (\mathbf{v}_i)_2\big) \in A'$. Note that this implies that for all i, $(\mathbf{v}_i)_1 = (\mathbf{v}_i)_2$. Consider the tuple $\big(f((\mathbf{v}_1)_1, (\mathbf{v}_2)_1, \cdots, (\mathbf{v}_n)_1), f((\mathbf{v}_1)_2, (\mathbf{v}_2)_2, \cdots, (\mathbf{v}_n)_2)\big) = \big(f((\mathbf{v}_1)_1, (\mathbf{v}_2)_1, \cdots, (\mathbf{v}_n)_1), f((\mathbf{v}_1)_1, (\mathbf{v}_2)_1, \cdots, (\mathbf{v}_n)_1)\big) \in B'$. Thus, f is a polymorphism of $(=,=)$ as well which implies that f is a polymorphism of Q. It has already been shown [Pip02b; BG21b; Bar+21] that if polymorphisms of a PCSP P are a subset of polymorphisms of Q, then Q is log space reducible to P. Thus, P and Q are equivalent under log space reductions. $\square$

Chapter 6

Robust Algorithms and SDPs for Promise CSPs

6.1 Introduction

Horn-SAT and 2-SAT are Boolean constraint satisfaction problems (CSPs) that admit simple combinatorial algorithms for satisfiability. They are both examples of *bounded width* CSPs, which means that the existence of locally consistent assignments (which satisfy all local constraints involving some bounded number of variables, and which are consistent on the intersections) implies the existence of a global satisfying assignment.[1]

While the simple local propagation algorithms for Horn-SAT and 2-SAT work when the instance is perfectly satisfiable, they are not robust to errors—if the given instance is almost satisfiable, i.e., $(1-\epsilon)$-satisfiable for $\epsilon \to 0$, the local consistency based algorithms do not guarantee solutions that satisfy almost all the constraints. In a beautiful work, Zwick [Zwi98] initiated the study of finding "robust" algorithms for Constraint Satisfaction Problems (CSPs), namely algorithms that output solutions satisfying $1 - f(\epsilon)$ fraction of the constraints when the instance is guaranteed to be $1-\epsilon$ satisfiable, where $f(\epsilon) \to 0$ as $\epsilon \to 0$. Zwick obtained robust algorithms for 2-SAT using SDP rounding and for Horn-SAT based on LP rounding. The PCP theorem together with Schaefer's reductions [Sch78] shows that Boolean CSPs that are NP-Hard are also APX-hard with perfect completeness, which in particular means that they do not admit robust satisfiability algorithms. The only other interesting Boolean CSP besides Horn-SAT and 2-SAT for which satisfiability is polynomial-time decidable is Linear Equations modulo 2. Håstad [Hås01] in his seminal work showed that even for 3-LIN (when all equations involve just three variables), for every $\epsilon, \delta > 0$, it is NP-Hard to output a solution satisfying $\frac{1}{2} + \delta$ fraction of the constraints even when the instance is guaranteed to have a solution satisfying $(1-\epsilon)$ fraction of the constraints.

Unlike Horn-SAT or 2-SAT, the satisfiability algorithm for 3-LIN is not local, and 3-LIN does not have bounded width. Thus, for Boolean CSPs, bounded width characterizes robust

[1] For CSPs, this is equivalent to solvability by $O(1)$ rounds of the Sherali Adams LP hierarchy.

satisfiability. For CSPs over general domains, a landmark result in the algebraic approach to CSP due to Barto and Kozik [BK14b] showed that CSPs that are not bounded width can express linear equations. A reduction from Håstad's result then shows that CSPs that are not bounded width do not admit robust algorithms. Guruswami and Zhou [GZ12] conjectured the converse—namely that all bounded width CSPs, over any domain, admit robust algorithms. Another work by Barto and Kozik [BK16] resolved this conjecture in the affirmative, thus giving a full characterization of CSPs that have robust algorithms.

In this work, we study robust algorithms for PCSPs. Broadly speaking, the study of PCSPs has been on two fronts: First, understanding which PCSPs can be solved in polynomial time, motivated by questions such as approximate graph coloring and $(2+\epsilon)$-SAT. Second, understanding the power of various algorithms for PCSPs. We initiate the study of robust algorithms for PCSPs motivated by both these directions. On one hand, robust algorithms are important on their own, understanding whether there are algorithms that work even with a small noise in the input. On the other hand, robust algorithms are equivalent to bounded width, $O(1)$ levels of Sherali Adams, and solvability by basic SDP for CSPs. The question of whether the same holds for PCSPs as well is a way to understand the power of these algorithmic tools themselves.

As is the case with CSPs, a natural approach to characterize which PCSPs have robust algorithms is via bounded width of PCSPs. However, it turns out that bounded width for PCSPs is weaker than having robust algorithms. Concretely, Atserias and Dalmau [AD22] have proved recently that the PCSP (1-in-3-SAT, NAE-3-SAT) does not have bounded width. As we shall show later, this PCSP indeed has a robust algorithm. Atserias and Dalmau also proved that this PCSP indeed can be solved by $O(1)$ levels of Sherali-Adams, and as we shall later, it can also be solved using the basic SDP.

Having ruled out the connection to bounded width, we study robust algorithms for PCSPs directly via their polymorphisms. Polymorphisms are closure properties of satisfying solutions to (Promise) CSPs. As a concrete example, consider the 2-SAT CSP: given an instance I of 2-SAT over n variables $x_1, x_2, \ldots, x_n$, suppose that $\mathbf{u}, \mathbf{v}, \mathbf{w}$ are three assignments to these variables satisfying all the constraints in I, then the assignment $\mathbf{z}$ that is coordinatewise Majority operation on three bits, i.e., $z_i = \mathrm{MAJ}(u_i, v_i, w_i)$ for every $i \in [n]$, also satisfies all the constraints in I. This shows that the Majority function on three variables, or more generally, any odd number of variables is a polymorphism of the 2-SAT CSP. Similarly, the Parity function on any odd number of variables is a polymorphism of 3-LIN, whereas there are no non-trivial polymorphisms for 3-SAT. Polymorphisms are the central objects in the *Universal algebraic approach* to CSPs [JCG97; Jea98; BJK05], which has then been extended to PCSPs [BG21b; Bar+21].

At a high level, the existence of non-trivial polymorphisms implies algorithms, and vice-versa. The key challenge is to precisely characterize which polymorphisms lead to algorithms. It is known that the polymorphism family of a PCSP fully captures its computational complexity i.e., if there are PCSPs Γ, Γ' such that the polymorphism family of Γ, $\mathrm{Pol}(\Gamma)$ is contained in $\mathrm{Pol}(\Gamma')$, then Γ' is formally easier than Γ, i.e., there is a *gadget reduction* from Γ' to Γ. It turns out that this gadget reduction also preserves the existence of robust algorithms. This leads to the following questions: *Which polymorphisms lead to robust algorithms for PCSPs? Can we use the* *polymorphism characterization of robust algorithms to relate Basic SDP and robust algorithms for PCSPs?*

We make progress on these questions on two fronts: first, for Boolean symmetric[2] PCSPs where we allow negation of variables, we study which polymorphisms lead to robust algorithms, and which Boolean symmetric PCSPs do not admit robust algorithms. Second, towards understanding the power of basic SDP for promise CSPs, we introduce a minion $\mathcal{M}$ and show that a PCSP Γ can be solved by basic SDP if and only if there is a minion homomorphism from $\mathcal{M}$ to the minon of polymorphisms of Γ.

As is the case with CSPs, if a PCSP is NP-Hard, then it does not admit robust algorithms. Thus, the above question is relevant only for PCSPs that can be solved in polynomial time. A large class of PCSPs for which polynomial time solvability has been fully characterized is the Boolean symmetric PCSPs. In [BG21b], the authors showed that for a Boolean symmetric PCSP Γ with folding i.e., we allow negating the variables (later this restriction was removed by [Fic+19]), Γ can be solved in polynomial time if and only if it contains at least one of Alternate-Threshold(AT), Majority (MAJ) or Parity polymorphisms of all odd arities. While AT and MAJ are noise stable functions, Parity is highly sensitive to noise i.e., if we perturb each input with a small probability, the function output changes significantly. In fact, Parity having low noise sensitivity can also be viewed as one reason why 3-LIN, despite having Parity as polymorphisms, does not admit robust algorithms. Thus, for Boolean symmetric folded PCSPs, a natural candidate characterization of robust algorithms is the existence of AT or MAJ polymorphisms.

On the algorithmic front, we prove that this indeed is the case, and on the hardness side, assuming the Unique Games Conjecture [Kho02a], we show that the absence of AT of MAJ polymorphisms implies the lack of robust algorithms.

6.1.1 Robust algorithms

Our main algorithmic result is the following.

Theorem 63. *Every Boolean folded PCSP Γ that contains AT or MAJ polymorphisms of all odd arities admits a polynomial time robust algorithm. In particular,*

1. *If Γ contains MAJ polymorphisms of all odd arities, then for every $\epsilon > 0$, there exists a polynomial time algorithm that given an instance of Γ that is promised to have a solution satisfying $1-\epsilon$ fraction of the constraints, outputs a solution satisfying $1-\tilde{O}(\epsilon^{\frac{1}{3}})$ fraction of the constraints.*[3]
2. *If Γ contains AT polymorphisms of all odd arities, then for every $\epsilon > 0$, there exists a polynomial time algorithm that outputs a solution satisfying $1-O\left(\frac{\log\log\frac{1}{\epsilon}}{\log\frac{1}{\epsilon}}\right)$ fraction of the constraints on an instance promised to have a solution satisfying $1-\epsilon$ fraction of the constraints.*

[2] A predicate P is symmetric if for every satisfying assignment $(x_1,\ldots,x_n)$ to P, any permutation of that assignment also satisfies P. For a Boolean predicate whether an assignment satisfies a predicate depends only on the Hamming weight. A PCSP is said to be symmetric if all the predicates in the template are symmetric.

[3] Here, $\tilde{O}$ hides multiplicative poly logarithmic factors.

We obtain our robust algorithms by rounding the *basic Semi Definite Programming(SDP)*[4] relaxation. For the Majority polymorphisms case, we use the same robust algorithm of Charikar, Makarychev, Makarychev [CMM09] for 2-SAT, with a completely different analysis. The exact version of 2-SAT has a simple algorithm based on rounding basic SDP: suppose that we have the predicate $x_1 \vee x_2$. We find vectors $\mathbf{v}_0$ and $\mathbf{v}_1, \mathbf{v}_2$ that satisfy the basic SDP constraints. As the basic SDP has zero error, we get that $\langle \mathbf{v}_1, \mathbf{v}_0 \rangle + \langle \mathbf{v}_2, \mathbf{v}_0 \rangle \geq 0$. This gives a simple rounding algorithm: we round x_i to True if and only if $\langle \mathbf{v}_i, \mathbf{v}_0 \rangle \geq 0$. This is not a robust algorithm: when there is a weaker guarantee that $\langle \mathbf{v}_1, \mathbf{v}_0 \rangle + \langle \mathbf{v}_2, \mathbf{v}_0 \rangle \geq -\epsilon$, the above rounding can round both variables to False. Zwick [Zwi98] gave the first robust algorithm for 2-SAT where he does "outward rotation" where he rotates the vectors $\mathbf{v}$ with $|\langle \mathbf{v}, \mathbf{v}_0 \rangle| \leq \epsilon^{\frac{1}{3}}$ by certain angle depending on $\langle \mathbf{v}, \mathbf{v}_0 \rangle$ before using the above algorithm. [CMM09] gave an algorithm that does the update continuously instead of a sudden jump at a fixed threshold, that gave an optimal error $O(\sqrt{\epsilon})$. We use the same algorithm as theirs, but the analysis is completely different as we need our analysis to be predicate independent, and just use the existence of MAJ polymorphisms as a black box.

As a concrete example, consider the PCSP ($\geq$ 2-in-4-SAT, 4-SAT). Here, we are given a 4-SAT instance in which there is an assignment where in at least $1-\epsilon$ fraction of the constraints, there are at least two literals that are set to be true. Under this promise, can we find an assignment in polynomial time where in at least $1 - f(\epsilon)$ fraction of the constraints, at least one literal is set to be true, for some function f with $f(\epsilon) \to 0$ as $\epsilon \to 0$? The analysis of [CMM09] for the 2-SAT problem where they compute the probability of a two dimensional Gaussian with a given mean and covariance matrix lying in the positive orthant, cannot be easily extended to 4-dimensions. Instead, we follow a simpler analysis where we choose a single variable in the constraint carefully and show that with high probability, it gets rounded to True, and thus the predicate is satisfied. For a general predicate (P, Q), we pick the coordinate by first showing that there is a halfspace separating the convex hull of P using the fact that (P, Q) contains Majority of all odd arities as polymorphisms. While our analysis is simpler and more general than [CMM09], we only achieve an error parameter of $\tilde{O}\big(\epsilon^{1/3}\big)$, similar to [Zwi98].

For the Alternating-Threshold (AT) case, we combine these ideas with a random geometric sampling trick. As a concrete example, consider the PCSP (1-in-3-SAT, NAE-3-SAT). For the exact case, we can solve the problem using basic SDP via hyperplane rounding: The vectors $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ satisfy the property that the sum $\mathbf{v}_s = \mathbf{v}_1 + \mathbf{v}_2 + \mathbf{v}_3$ is along the direction of $\mathbf{v}_0$. Thus, we pick a hyperplane with a normal vector $\mathbf{r}$ that is orthogonal to $\mathbf{v}_0$, and round $\mathbf{v}_i$ to be True if $\langle \mathbf{v}_i, \mathbf{r} \rangle > 0$, and False otherwise. For the robust setting, we get that the vector $\mathbf{v}_s$'s component normal to $\mathbf{v}_0$ is small. Using this, we design a rounding scheme that is similar to the above, with the addition that when the vector $\mathbf{v}_i$'s component normal to $\mathbf{v}_0$ is small enough, we round it to True or False depending on its component along $\mathbf{v}_0$. The final ingredient is a geometric sampling trick where we sample the ratio of these two metrics randomly from a geometric series.

[4] Basic SDP(formally described in Section 6.3) is a well studied [Rag08] SDP relaxation of CSPs.

6.1.2 Unique Games based hardness

Unlike the algorithms part, in our hardness results, we crucially use the symmetry of the predicates. Furthermore, we assume that the PCSP contains a single predicate pair $\Gamma = (P, Q)$ that does not admit AT or MAJ polymorphisms, and we allow constraints to use negations of variables and unary constraints that set variables to be True or False. This is equivalent to asserting that the associated polymorphisms are folded and idempotent.[5] We show that for these Boolean symmetric folded idempotent PCSPs without AT and MAJ polymorphisms, the basic SDP relaxation has an integrality gap with perfect completeness, which by Raghavendra's framework connecting SDP gaps and Unique-Games hardness [Rag08], rules out robust satisfaction algorithms (under the Unique Games conjecture [Kho02a]).

More formally, we state our main hardness result below.

Theorem 64. *For every Boolean symmetric folded idempotent PCSP $\Gamma = (P, Q)$ such that $\mathrm{AT}_{L_1}, \mathrm{MAJ}_{L_2} \notin \mathrm{Pol}(\Gamma)$ for some odd integers L_1, L_2, Γ does not admit a robust algorithm assuming the Unique Games Conjecture.*

As mentioned above, our Unique Games hardness (Theorem 64) is based on an integrality gap for the basic SDP relaxation. Towards this end, we present a general recipe for showing integrality gaps with respect to basic SDP for Promise CSPs via colorings of the n-dimensional unit sphere.

Consider a simple local rounding scheme for a PCSP $\Gamma = (P, Q)$: for every $n \geq 1$, there is a fixed rounding function $f_n : \mathbb{S}^n \to \{-1, +1\}$. We consider the basic SDP relaxation of Γ and obtain $\mathbf{v}_x \in \mathbb{R}^n$ corresponding to every variable x. We round it to an integral solution where we map $\mathbf{v}_x$ to $f_n(\mathbf{v}_x)$. For this to be a valid rounding scheme, whenever a configuration of vectors $V = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k)$ can be assigned to the variables in a constraint, the corresponding integral values $(f_n(\mathbf{v}_1), f_n(\mathbf{v}_2), \ldots, f_n(\mathbf{v}_k))$ must belong to Q. Note that proving that there are no such local rounding functions is a necessary step towards showing integrality gaps for the PCSP Γ. By using compactness arguments, we can show that this is sufficient as well.

As a concrete example, consider the CSP $P = 3$-LIN: a set of three vectors $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ can be assigned to a set of vertices x_1, x_2, x_3 of a constraint by the basic SDP if the gram matrix of these vectors is in the convex hull of the gram matrices of the satisfying assignments to P. We refer to such a set of three vectors as a P-configuration. For the 3-LIN case, this condition can be translated to the fact that the three vectors are pairwise orthogonal. Thus, to show that basic SDP does not solve 3-LIN, it suffices to show that for some integer n, there is no function $f : \mathbb{S}^n \to \{-1, +1\}$ such that whenever $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3 \in \mathbb{S}^n$ are mutually orthogonal, then there are odd number of $+1$s in $(f(\mathbf{v}_1), f(\mathbf{v}_2), f(\mathbf{v}_3))$. As we allow negation of variables, we also require such a function f to be folded, i.e., $f(-\mathbf{v}) = -f(\mathbf{v})$ for every $\mathbf{v} \in \mathbb{S}^n$. In fact, it's easy to show such a coloring f does not exist: consider a set of three mutually orthogonal vectors $V = (\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3)$ and their negations, $V' = (-\mathbf{v}_1, -\mathbf{v}_2, -\mathbf{v}_3)$. Note that both these are a valid P-configurations, but at least one of $(f(\mathbf{v}_1), f(\mathbf{v}_2), f(\mathbf{v}_3))$, $(f(-\mathbf{v}_1), f(-\mathbf{v}_2), f(-\mathbf{v}_3))$ must have an even number of $+1$s, completing the proof that there is no such local rounding function. This corresponds to the textbook Basic SDP integrality gap instance for 3-LIN consisting of the

[5] We say that a function $f : \{-1, +1\}^n \to \{-1, +1\}$ is folded if $f(-\mathbf{x}) = -f(\mathbf{x})$. We say f is idempotent if $f(b, b, \ldots, b) = b$ for every $b \in \{-1, +1\}$.

constraints $\{(x_1, x_2, x_3), (\overline{x}_1, \overline{x}_2, \overline{x}_3)\}$.

We can define the P-configurations for an arbitrary PCSP $\Gamma = (P, Q)$, and use this approach to show the absence of sphere coloring "respecting" Γ, which in turn implies the required integrality gap with respect to the basic SDP for Γ. While the P-configurations in the above proof for 3-LIN are easy to study, in general, proving the absence of sphere coloring is challenging. For example, consider the PCSP (P, Q) where $P = \{1, 5\}$-in-5-SAT, $Q = \{1, 2, 3, 4, 5\}$-in-5-SAT. Here, a set of P-configurations are five unit vectors such that every two distinct vectors have inner product equal to $\frac{-1}{5}$. The sphere coloring problem is then to show that there exists n such that for any folded $f : \mathbb{S}^n \to \{-1, +1\}$, there exists a set of five vectors in $\mathbb{S}^n$ with every pair of them having inner product equal to $\frac{-1}{5}$ that are all colored -1.

Such problems where the goal is to find a monochromatic structure in sphere colorings are studied under the title "sphere Ramsey theory". In a striking result using tools from combinatorics, linear algebra, and Banach space theory, Matoušek and Rödl [MR95] proved that every set of affinely independent vectors V whose circumradius is smaller than 1 is sphere Ramsey—i.e., for every r, there exists n large enough such that every r-coloring of $\mathbb{S}^n$ must have a monochromatic set U that is congruent to V. This directly answers the above question regarding sphere coloring of (P, Q) where $P = \{1, 5\}$-in-5-SAT, $Q = \{1, 2, 3, 4, 5\}$-in-5-SAT.

For an arbitrary Boolean symmetric PCSP $\Gamma = (P, Q)$, to prove Theorem 64, we first reduce the problem into a fixed number of templates using the properties of AT and MAJ polymorphisms [BG21b]. Then, we use the result of Matoušek and Rödl [MR95], and a connectivity lemma for configurations to show the absence of sphere colorings for these templates, except for Γ^* mentioned in Theorem 64.

Our results highlight a close connection between robust algorithms for PCSPs and being solved by the basic SDP. For a Promise CSP (P, Q), by being solved by the basic SDP, we mean that for every instance I of a PCSP Γ has an integral solution satisfying Q if and only if the basic SDP relaxation using P is feasible on I. In other words, we can use the basic SDP to solve the decision version of the PCSP Γ. As our algorithms for the AT and MAJ polymorphisms are based on rounding the basic SDP, we get that every Boolean folded PCSP that contains AT or MAJ polymorphisms can be solved by the basic SDP. In the proof of Theorem 64, we showed that a vast majority of Boolean symmetric folded PCSPs without AT or MAJ polymorphisms cannot be solved by the basic SDP. This suggests a more general relation between the basic SDP and robust algorithms for PCSPs. Informally, for both the existence of robust algorithms and being solved by the basic SDP, the underlying requirement is the existence of polymorphism families that are robust to noise. While our results show that this is true for the PCSPs that we study in this chapter, we believe this is a more general phenomenon.

Conjecture 65. *A PCSP Γ has a polynomial time robust algorithm if and only if Γ can be solved by the basic SDP.*

Note that if there is an integrality gap for Γ with respect to the basic SDP relaxation, then by Raghavendra's [Rag08] result, we get that Γ does not have a polynomial time robust algorithm, assuming the Unique Games Conjecture. This already proves one direction of Conjecture 65. The other direction is more interesting: can we obtain robust algorithms for PCSPs just using the fact that basic SDP solves them exactly? We also remark that the conjecture is already proven to be true for CSPs, where the existence of robust algorithms [BK16] and solvability by basic SDP [TŽ18] are both shown to be equivalent to having bounded width.

6.1.3 Minion characterization of basic SDP

In addition to our concrete characterization of robust algorithms for a subfamily of PCSPs, we also present a novel algebraic characterization of which PCSPs can be solved via basic SDPs. Originally, in the study of CSPs, such algebraic characterizations were structured as follows (e.g., [Bul17; Zhu20]).

- *"Algorithm $\mathcal{A}$ solves* $\mathrm{CSP}(\Gamma)$*, iff there is a polymorphism* $f \in \mathrm{Pol}(\Gamma)$ *with specific properties."*

A key property of the polymorphisms of a PCSP Γ is that one can take *minors* by identifying coordinates. Relationships between a finite set of functions can be captured by *identities*, but more advanced relationships can be captured by infinite objects known as *minions*.

Since the early days of PCSPs, it has been known that a single polymorphism cannot dictate hardness (c.f., [BG21b]), and thus one must instead consider a *sequence* of polymorphisms (e.g., [Bra+20]):

- *"Algorithm $\mathcal{A}$ solves* $\mathrm{PCSP}(\Gamma)$*, if and only if there is an infinite sequence of polymorphism* $f_1, f_2, \ldots \in \mathrm{Pol}(\Gamma)$ *with specific properties."*

However, in many cases, such a characterization is unfeasible or unwieldy. Instead a more general characterization, first characterized by [Bar+21], captures the structure of polymorphism via a *minion* (formally defined in Section 6.6).

- *"Algorithm $\mathcal{A}$ solves* $\mathrm{PCSP}(\Gamma)$*, if and only if there is minion homomorphism from* $\mathcal{M}_\mathcal{A}$ *to* $\mathrm{Pol}(\Gamma)$*."*

Many recent papers [Bra+20; CŽ22a; CŽ22b] have proven such characterizations in various contexts. Our contribution to this line of work is showing that the basic SDP can be captured by a minion, which we call $\mathcal{M}_{\mathrm{SDP}}$.

Theorem 66. *The exact*[6] *basic SDP solves* $\mathrm{PCSP}(\Gamma)$ *if and only if there is a minion homomorphism from* $\mathcal{M}_{\mathrm{SDP}}$ *to* $\mathrm{Pol}(\Gamma)$.

We note that the theorem applies equally to Boolean and non-Boolean PCSPs.

The construction of the $\mathcal{M}_{\mathrm{SDP}}$ minion is inspired by the vector interpretation of solutions to the Basic SDP. Each object in the minion is a collection of orthogonal vectors which sum to a reference vector $\mathbf{v}_0$. The minors involve adding groups of vectors together. Having a minion homomorphism from $\mathcal{M}_{\mathrm{SDP}}$ to $\mathrm{Pol}(\Gamma)$ implies that there are polymorphisms of Γ whose minors behave exactly like combining orthogonal vectors.

Proving Theorem 66 has a few technical hurdles. One challenge is that SDP solutions may require vectors of an arbitrarily large dimension. In order for these arbitrarily-large dimensional relationships to be captured in our minion, we have that the families of vectors making up $\mathcal{M}_{\mathrm{SDP}}$

[6]Here 'exact' means that we verify that the basic SDP is solved to exact precision, whereas $\mathrm{poly}(n)$ bits of precision is the computational limit. A more thorough discussion of this technicality is in Section 6.6.

reside in a (countably) infinite-dimensional vector space. Similar techniques have been used in other minion constructions [CŽ22a; CŽ22b].

Another challenge that appears specifically unique to our work is that a Basic SDP solution gives a vector corresponding to each variable, but in order for the proof to go through additional vectors are needed which correspond to the clauses. [The variable vectors are ”projections” of the clause vectors.] Obtaining such clauses would typically be done via Sum-of-Squares or a related routine, but we prove that including such vector clauses are without loss of generality. That is, for any Basic SDP solution, it can be extended to a solution which includes clause vectors without modifying the original Basic SDP solution. This gives us enough vector structure to prove that the minion homomorphism corresponds to the Basic SDP solution.

Path to integrality gaps. As a direct consequence of the minion structure theorem, we can connect sphere coloring with integrality gaps. By a result of [Bar+21], the minion homomorphism $\mathcal{M}_{\mathrm{SDP}} \to \mathrm{Pol}(\Gamma)$ is equivalent to finding a satisfying assignment to a "universal" instance of $\mathrm{PCSP}(\Gamma)$ known as a *free structure*. In the case that Γ is a Boolean PCSP, this free structure for $\mathcal{M}_{\mathrm{SDP}}$ turns out an instance where every possible unit vector is a variable. The clauses correspond to collections of vectors which satisfying the corresponding basic SDP. For the general theory of approximation of basic SDPs, similar constructs with sphere coloring being a 'universal' gap have appeared in the literature ([Bra+21]). To make the integrality gaps more self-contained, we streamline the connection between sphere coloring and integrality gaps in Lemma 86.

Organization. We first start by introducing formal definitions and some general observations in Section 6.2 and Section 6.3. We provide our algorithmic results (Theorem 63) in Section 6.4 and prove the hardness results (Theorem 64) in Section 6.5. Finally, we study the basic SDP minion in Section 6.6.

6.2 Preliminaries

Notations. We use $[n]$ to denote the set $\{1, 2, \ldots, n\}$. For a k-ary relation $A \subseteq [q]^k$, we abuse the notation and use A both as a subset of $[q]^k$, and also as a predicate $A : [q]^k \to \{0, 1\}$. For a vector $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in \{-1, +1\}^n$, we use $\mathsf{hw}(\mathbf{x})$ to denote the number of $+1$s in $\mathbf{x}$, i.e., $\mathsf{hw}(\mathbf{x}) = \frac{n+\sum_{i=1}^n x_i}{2}$. For $S \subseteq \{0, 1, \ldots, k\}$, we use $\mathrm{Ham}_k\, S$ to denote $\{\mathbf{x} \in \{-1, +1\}^k : \mathsf{hw}(\mathbf{x}) \in S\}$. We use NAE_k to denote the set $\mathrm{Ham}_k\{1, 2, \ldots, k-1\}$. For vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, we use $\mathbf{x} \cdot \mathbf{y}$ and $\langle \mathbf{x}, \mathbf{y} \rangle$ interchangeably to denote $\sum_i x_i y_i$.

Boolean symmetric folded PCSPs. We restrict ourselves to Boolean symmetric folded PCSPs in this chapter.

Definition 67 (Boolean symmetric folded PCSP). *A PCSP $\Gamma = (A_1, B_1), (A_2, B_2), \ldots, (A_l, B_l)$ over a pair of domains D_1, D_2 is said to be Boolean symmetric folded if the following hold:*

1. *(Boolean) The domains D_1 and D_2 are both equal to $\{-1, +1\}$.*
2. *(Symmetric) All the relations are symmetric i.e., for every $i \in [l]$, and $\boldsymbol{x}, \boldsymbol{y}$ such that $\mathsf{hw}(\boldsymbol{x}) = \mathsf{hw}(\boldsymbol{y})$, we have $\boldsymbol{x} \in A_i$ if and only if $\boldsymbol{y} \in A_i$, and similarly, $\boldsymbol{x} \in B_i$ if and only if $\boldsymbol{y} \in B_i$.*

3. *(Folded) We allow negating the variables i.e., there exists $i \in [l]$ such that $A_i = B_i = \{(-1,+1),(+1,-1)\}$.*

AT **and** MAJ **polymorphisms.** We extensively study Alternate-Threshold (AT) and Majority (MAJ) polymorphisms in this chapter:

1. For an odd integer $L \geq 1$ and $\mathbf{x} \in \{-1,+1\}^L$, we have

$$\mathrm{AT}_L(\mathbf{x}) = \begin{cases} +1, & \text{if } x_1 - x_2 + x_3 - \ldots + x_L > 0 \\ -1, & \text{otherwise.} \end{cases}$$

2. For an odd integer $L \geq 1$ and $\mathbf{x} \in \{-1,+1\}^L$, we have

$$\mathrm{MAJ}_L(\mathbf{x}) = \begin{cases} +1, & \text{if } x_1 + x_2 + \ldots + x_L > 0 \\ -1, & \text{otherwise.} \end{cases}$$

We also use $\mathrm{AT}_L(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_L)$ for $\mathbf{x}_i \in \{-1,+1\}^k$ (similarly for MAJ) when applying AT_L coordinatewise. For a predicate $P \subseteq \{-1,+1\}^k$, we use $\mathrm{AT}_L(P)$ to denote the set $\bigcup_{\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_L \in P} \mathrm{AT}_L(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_L)$. We say that a AT(and resp. MAJ) is in $\mathrm{Pol}(\Gamma)$ if AT_L (and resp. MAJ_L) is in $\mathrm{Pol}(\Gamma)$ for every odd integer $L \geq 1$. For a predicate $P \subseteq \{-1,+1\}^k$, we use $O_{\mathrm{AT}}(P)$ (and similarly $O_{\mathrm{MAJ}}(P)$) to denote the set $\bigcup_{L \in \mathbb{N}, \text{ odd}} \mathrm{AT}_L(P)$.

Relaxations of PCSPs. We say that a PCSP Γ' is a relaxation of another PCSP Γ if $\mathrm{Pol}(\Gamma) \subseteq \mathrm{Pol}(\Gamma')$. If Γ' is a relaxation of Γ, then there is a *gadget reduction* from Γ' to Γ. More formally, it is referred to as a *positive primitive promise reduction(ppp-reduction)* from Γ' to Γ, or equivalently, as Γ' is *ppp-definable* from Γ.

Definition 68 (ppp-definability of PCSPs([BG21b])). *We say that a PCSP $\Gamma' = (P', Q')$ containing a single pair of predicates of arity k is ppp-definable from a PCSP Γ over the same domain pair if there exists a fixed constant l and a PCSP instance Ψ using Γ (we also allow identifying variables together) over $k + l$ variables $x_1, x_2, \ldots, x_k, y_1, y_2, \ldots, y_l$ such that*

1. *If $(x_1, x_2, \ldots, x_k) \in P'$, then there exist $y_1, y_2, \ldots, y_l$ such that $(x_1, x_2, \ldots, x_k, y_1, y_2, \ldots, y_l)$ satisfies the strong constraints in Γ.*
2. *If there exists a satisfying assignment $(z_1, z_2, \ldots, z_{k+l})$ to the weak constraints in Γ, then $(z_1, z_2, \ldots, z_k) \in Q'$.*

More generally, we say that Γ' is ppp-definable from Γ if every predicate pair in Γ' is ppp-definable from Γ. Brakensiek and Guruswami [BG21b] showed that if Γ' is a relaxation of Γ, then Γ' is ppp-definable from Γ. As the ppp-reductions are polynomial time reductions, this shows that Γ' can be reduced to Γ in polynomial time. We show that the ppp-reductions also preserve the existence of robust algorithms.

Lemma 69. *Suppose that the PCSP Γ' over a pair of domains D_1, D_2 is a relaxation of Γ over the same domain pair i.e., $\mathrm{Pol}(\Gamma) \subseteq \mathrm{Pol}(\Gamma')$. If Γ has a polynomial time robust algorithm, then Γ' has a polynomial time robust algorithm as well.*

Proof. As proved in [BG21b], Γ' can be ppp-reduced to Γ. Given an instance I' of Γ' over a set of variables U, we output an instance I of Γ containing $|U|$ original variables and a set of dummy variables. For every constraint Ψ' using (P', Q') involving the variables $u_1, u_2, \ldots, u_k$ in I', we have a set of dummy variables $v_1, v_2, \ldots, v_l$ and a set of constraints Ψ using Γ among $\{u_1', u_2', \ldots, u_k', v_1, v_2, \ldots, v_l\}$ as in Definition 68. Let $V' = U' \cup V$ denote the set of variables of I, where U' denotes the set of $|U|$ original variables corresponding to U, and V are the set of dummy variables. We claim that this reduction preserves robust algorithms.

1. (Completeness). Suppose that there exists an assignment $\chi' : U \to D_1$ to I' satisfying all the constraints. Then, there is an assignment $\chi : V \to D_1$ to the dummy variables which together with assigning χ' to the original variables satisfies all the constraints in I.
2. (Soundness). Suppose that there is an assignment $\chi : V' \to D_2$ satisfying $1 - \epsilon$ fraction of the constraints in I. As each dummy constraint set used $O(1)$ constraints, we get that in at least $1 - O(\epsilon)$ constraint sets, all the constraints Ψ are satisfied. This shows that the assignment χ restricted to U' satisfies $1 - O(\epsilon)$ fraction of the constraints in I'.

□

Elementary properties of Gaussians. We prove a couple of elementary properties of Gaussian distribution that we use later. First, we prove the following anti-concentration inequality for the standard Gaussian random variable.

Proposition 70. *Suppose that $X \sim \mathcal{N}(0,1)$ has the standard Gaussian distribution. Then, for every $\epsilon \geq 0$,*

$$\textit{Pr}\left(|X| \leq \epsilon\right) \leq O(\epsilon).$$

Proof. We have

$$\Pr\left(|X| \leq \epsilon\right) = \int_{-\epsilon}^{+\epsilon} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx \leq \int_{-\epsilon}^{+\epsilon} \frac{1}{\sqrt{2\pi}} dx = O(\epsilon).$$

□

We also state the following concentration inequality for 1-dimensional Gaussian.

Proposition 71. *Suppose that $X \sim \mathcal{N}(0, \sigma^2)$ has Gaussian distribution with variance σ^2. Then, for every $t \geq 0$,*

$$\textit{Pr}\left(X \geq t\right) \leq e^{-\frac{t^2}{2\sigma^2}}.$$

6.3 General Observations

6.3.1 Basic SDP setup

For simplicity when analyzing SDP rounding, we shall use $\{-1, 1\}$ as our Boolean domain. If it helps, think of 1 as 'True' and -1 as 'False.' We shall use bold text for vectors in real space, but non-bold text for PCSP tuples.

For $x \in \{-1, 1\}^k$, we let $\mathbf{v}_x \in \mathbb{R}^{k+1}$ be the column vector whose first coordinate is 1 and the remaining k coordinates are x. We define the KZ vertex [KZ97] at x to be $M_x := \mathbf{v}_x \mathbf{v}_x^T$. For

$P \subseteq \{-1,1\}^k$, we let $\mathrm{KZ}(P)$ denote the convex closure of $\{M_x : x \in P\}$. For $\epsilon > 0$, let $\mathrm{KZ}_\epsilon(P)$ denote the convex closure of $\mathrm{KZ}(P) \cup \{\epsilon M_x : x \in \{-1,1\}^k\}$ (that is you have ϵ error).

For an instance Φ of a PCSP Γ on n variables and m clauses, we let (A_i, B_i) be the clause type for the ith clauses and let $S_i \subseteq [n]$ be the subset of variables to which the ith clause is applied.[7]

The Basic SDP:[8]

$$\begin{aligned}
\textbf{minimize: } & \sum_{i=1}^{m} \epsilon_i \\
\textbf{subject to: } & M \succeq 0 \\
& \forall i \in \{0,1,\ldots,n\}, M_{i,i} = 1 \\
& \forall i \in [m], M|_{S_i \times S_i} \in \mathrm{KZ}_{\epsilon_i}(A_i).
\end{aligned}$$

We say that basic SDP is feasible on Φ if the above objective function is zero on Φ. We say that the basic SDP *solves* the PCSP Γ if for every Φ such that the basic SDP is feasible on Φ, there is an assignment from $\{-1,+1\}^n$ to the variables that satisfies all the m clauses with respect to the weaker constraints B_is.

6.3.2 Generic RHS reduction to "not x'

Let $\mathcal{F}$ be a family of functions (e.g., MAJ of all odd arities). For all $x \in \{-1,1\}^k$, let $Q_{\setminus x}$ be shorthand for $\{-1,1\}^k - \{x\}$.

Claim 72. *An algorithm $\mathcal{A}$ is a robust algorithm for* $\mathrm{InvPol}(\mathcal{F})$[9] *if and only if for all $k \geq 1$, $P \subseteq \{-1,1\}^k$ and $x \in \{-1,1\}^k \setminus O_{\mathcal{F}}(P)$, we have that $\mathcal{A}$ is a robust algorithm for* $\mathrm{PCSP}(P, Q_{\setminus x})$.

Proof. The "only if" direction is trivial as $(P, Q_{\setminus x}) \in \mathrm{InvPol}(\mathcal{F})$.

For the "if" direction, consider an instance and replace every clause of the form (A, B) with at most 2^k clauses of the form $\{(A, Q_{\setminus x}) : x \notin B\}$ all on the same set of variables. By assumption, $\mathcal{A}$ is a $1-\epsilon$ vs $1 - f(\epsilon)$ robust algorithm for these $(A, Q_{\setminus x})$ clauses, where f is the maximum of the tradeoff functions for all $(A, Q_{\setminus x})$. It is easy to see that the assignment $\mathcal{A}$ produces is a $1-\epsilon$ vs $1 - 2^k f(\epsilon)$ algorithm for $\mathrm{PCSP}(A, B)$, which is robust as 2^k depends only on the choice of template, which is independent of ϵ. $\square$

6.4 Robust Algorithms

6.4.1 CMM is a robust algorithm for MAJ

We show that the robust algorithm of Charikar, Makarychev, and Makarychev [CMM09] for 2-SAT generalizes to every PCSP Γ that has Majority polymorphisms of all odd arities. First, we recall the algorithm.

[7]Technically S_i is a "subtuple" not a subset as order matters, but how this is handled will always be obvious.

[8]This formulation may not make it clear why this is in fact an SDP (i.e., the dependences in ϵ_i is linear), but this is the most compact way to say what is really going on.

[9]We define $\mathrm{InvPol}(\mathcal{F})$ to be the set of *finite* promise templates Γ with $\mathcal{F} \subseteq \mathrm{Pol}(\Gamma)$.

1. Given an instance of Γ containing n variables, solve the basic SDP and obtain the Gram matrix $M \in \mathbb{R}^{(n+1)\times(n+1)}$. Let μ be the 0th column of M minus the first entry. Let Σ be the lower-right $[n] \times [n]$ submatrix of M.
2. Sample an n dimensional Gaussian $\zeta \sim \mathcal{N}(\mathbf{0}, \Sigma)$. (Note that Σ is PSD.)
3. Set[10] $\gamma = (\epsilon)^{\frac{2}{3}}$.
4. For each $i \in [n]$, round as follows

$$x_i = \begin{cases} +1 & \zeta_i \geq -\mu_i/\gamma \\ -1 & \text{otherwise.} \end{cases}$$

We shall prove the following analysis guarantee about our algorithm.

Theorem 73. *Let Γ be a PCSP such that $\mathrm{MAJ} \in \mathrm{Pol}(\Gamma)$. Let Ψ be an instance of $\mathrm{PCSP}(\Gamma)$ for which there is a basic SDP solution with a completeness of $1-\epsilon$ (i.e., the error value of the SDP solution is ϵm, where m is the number of clauses). Then, the CMM algorithm above find an 'integral' assignment to Ψ which satisfies $1 - \tilde{O}_\Gamma(\epsilon^{1/3})$ fraction of the clauses in expectation.*[11]

We analyze the algorithm clause by clause. Fix a clause using the predicate pair (P, Q). Let k denote their arity i.e., $P, Q \subseteq \{-1,+1\}^k$. Suppose that the basic SDP solution satisfies the constraint with probability $1-c$ i.e., the local distribution is supported with $(1-c)$ fraction of the predicates from P. Our goal is to upper bound the probability that the rounded solution violates the constraint Q by a function of ϵ and c. Using the fact that the expected value of c over all the constraints is at most ϵ, we get our required robustness guarantee. More formally, we prove the following

Lemma 74. *Let (P, Q) be a clause in the instance Ψ. Presume the basic SDP gives P a value of $1-c$, the probability that Q is satisfied by the CMM algorithm is*

$$O_\Gamma\left(\sqrt{\epsilon} + \sqrt{(\gamma\sqrt{\log\frac{1}{\epsilon}} + 2c)\log\frac{1}{\epsilon}} + \frac{c}{\gamma}\right).$$

As the expected value of c over all the constraints is at most ϵ, and using Jensen's, we get that the net error probability is $\tilde{O}\left(\epsilon^{1/3}\right)$, proving Theorem 73. Thus, it suffices to prove Lemma 74.

By Claim 72, it suffices to find a robust algorithm for $\mathrm{PCSP}(P, Q_{\setminus x})$ where $x \notin O_{\mathrm{MAJ}}(P)$.

Observe that CMM is "sign-symmetric" in the following sense: if a variable is replaced by its negation then the probabilities that variable are assigned ± 1, exactly interchange. Furthermore, replacing a variable with its negation in a clause (P, Q) does not change whether $\mathrm{MAJ} \in \mathrm{Pol}(P, Q)$. Henceforth, we may assume that $Q = \{-1,+1\}^k \setminus \{-1,-1,\ldots,-1\}$ (i.e., $x = (-1,-1,\ldots,-1)$).

Let $\mathcal{P}$ be the convex hull of P, where the tuples are viewed as vectors in $\mathbb{R}^k$. We prove the following property about $\mathcal{P}$ using the fact that the PCSP (P, Q) has Majority of all odd arities as polymorphisms.

[11] We use O_Γ to denote a hidden constant which depends on the specific template Γ.

Lemma 75. *Let $P \subseteq \{-1,+1\}^k$ be a predicate such that $(-1,-1,\ldots,-1) \notin O_{\mathrm{MAJ}}(P)$. Then, there is a hyperplane separating $\mathcal{P}$ from the origin: there exists $\boldsymbol{w} \in \mathbb{R}^k$, $\boldsymbol{w} \geq 0$ and $\|w\|_1 = 1$ such that for every $\boldsymbol{a} \in \mathcal{P}$, $\langle \boldsymbol{a}, \boldsymbol{w} \rangle \geq 0$.*

Proof. Consider the following linear program,

$$\begin{aligned} \textbf{maximize: } & \eta \\ \textbf{subject to: } & \sum_{i=1}^{k} w_i = 1 \\ & \forall a \in P, \eta - \sum_{i=1}^{k} a_i w_i \leq 0 \\ & \mathbf{w} \geq 0 \end{aligned}$$

It suffices to prove that the objective of this linear program is non-negative. To do this, we consider the dual program on variables $\nu \in \mathbb{R}$ and $\lambda_a \in \mathbb{R}$ for $a \in P$:

$$\begin{aligned} \textbf{minimize: } & \nu \\ \textbf{subject to: } & \lambda \geq 0 \wedge \sum_{a \in P} \lambda_a = 1 & \text{(dual of } \eta) \\ & \forall i \in [k], \nu - \sum_{a \in P} a_i \lambda_a \geq 0 & \text{(dual of } \mathbf{w}) \end{aligned}$$

As all the coefficients used in the LP are rational, we may assume that ν and λ are rational. Assume for sake of contradiction that there is a solution to the dual LP with $\nu < 0$. Then, we have that for all $i \in [k]$,

$$\sum_{a \in P} a_i \lambda_a < 0.$$

Let N be the least common denominator of the λ_a's. Consider the set of satisfying assignments to P where we take $2N\lambda_a$ copies of a for each $a \in P$. We also add an arbitrary element b of a to our set. As $\sum_{a \in P} a_i \lambda_i N < 0$ for every $i \in [k]$, and $\sum_{a \in P} a_i \lambda_i N$ is an integer, we get that for every $i \in [k]$, $2\sum_{a \in P} a_i \lambda_a N + b \leq 0$. In other words, when we apply MAJ_{2N+1} coordinatewise to this set of assignments in P, we get $(-1,-1,\ldots,-1)$. As (P,Q) contains Majority of all odd arities as polymorphisms, this implies that the resulting output $(-1,-1,\ldots,-1)$ is in Q, a contradiction.

Thus, the objective η of the original LP is non-negative, completing the proof. □

Now we use this lemma to complete the proof of Lemma 74. Suppose that the basic SDP solution satisfies the constraint (P,Q) with error equal to c i.e., there is a local distribution of $\{-1,+1\}^k$ that supports the vectors such that the weight of the assignments not in P is at most c. Let $K = 2^k$. We have probabilities $p_1, p_2, \ldots, p_K$ corresponding to the K local assignments $a_1, a_2, \ldots, a_K \in \{-1,+1\}^k$ where each $p_i \geq 0$ and $\sum_{i \in [K]} p_i = 1$ such that

$$\sum_{i \in [K], a_i \in P} p_i = 1 - c$$

Using Lemma 75, we get $\mathbf{w} \in \mathbb{R}^k$ with $\mathbf{w} \geq 0$ and $\|\mathbf{w}\|_1 = 1$ such that $\mathbf{w}^T a_i \geq 0$ for all $a_i \in P$. Combining this with the above properties of the basic SDP solution, we get the following.

1. (First moment). We have

$$
\begin{aligned}
\mathbf{w}^T \mu &= \sum_{i \in [K]} p_i \mathbf{w}^T a_i \\
&\geq -c \text{ (Using Lemma 75 and } -1 \leq \mathbf{w}^T a_i \leq 1 \, \forall i \in [K] \text{)}
\end{aligned}
$$

2. (Second moment). We have

$$
\mathbf{w}^T \Sigma \mathbf{w} = \sum_{i \in K} p_i (\mathbf{w}^T a_i)^2 \leq \sum_{i \in K, a_i \in P} p_i (\mathbf{w}^T a_i)^2 + c \leq \sum_{i \in K, a_i \in P} p_i \mathbf{w}^T a_i + c \leq \mathbf{w}^T \mu + 2c\,.
$$

We do casework on the value of $\mathbf{w}^T \mu$. First, consider the case that $\mathbf{w}^T \mu \geq \kappa = \gamma \sqrt{\log \frac{1}{\epsilon}}$. As $\|\mathbf{w}\|_1$=1, and $\mathbf{w} \geq 0$, there exists $i \in [k]$ such that $\mu_i \geq \kappa$. As $\zeta_i \sim \mathcal{N}(0,1)$, using Proposition 71, with probability at least $1 - \sqrt{\epsilon}$, we have $\zeta_i \geq -\frac{\mu_i}{\gamma}$. Thus, with probability at least $1 - \sqrt{\epsilon}$, the rounded solution satisfies Q.

Henceforth, we assume $\mathbf{w}^T \mu < \kappa$. For notational convenience let $\mathbf{t} = -\mu / \sqrt{\epsilon}$. We have

$$
\mathbf{w}^T \mathbf{t} \leq \frac{c}{\gamma} \tag{6.1}
$$

and $\mathbf{w}^T \Sigma \mathbf{w} \leq \kappa + 2c$. Note that $\mathbf{w}^T \zeta \sim \mathcal{N}(0, \mathbf{w}^T \Sigma \mathbf{w})$. Thus, using Proposition 71, with probability at least $1 - \sqrt{\epsilon}$, we have that

$$
|\mathbf{w}^T \zeta| \leq O\left(\sqrt{(\kappa + 2c) \log \frac{1}{\epsilon}} \right) \tag{6.2}
$$

Note that the rounded solution does not satisfy Q only if $\mathbf{t} \geq \zeta$. We now upper bound the probability that this can occur. Together with Equation (6.1) and Equation (6.2), $\mathbf{t} \geq \zeta$ implies that

$$
0 \leq \mathbf{w}^T (\mathbf{t} - \zeta) \leq O\left(\sqrt{(\kappa + 2c) \log \frac{1}{\epsilon}} + \frac{c}{\gamma} \right).
$$

Take some coordinate with $w_i \geq 1/k$ and note that

$$
\mathbf{t}_i - \zeta_i \in \left[0, O\left(\sqrt{(\kappa + 2c) \log \frac{1}{\epsilon}} + \frac{c}{\gamma} \right) \right],
$$

but this can only happen with probability $O\left(\sqrt{\kappa + c} \log \frac{1}{\epsilon} + \frac{c}{\sqrt{\epsilon}} \right)$ using Proposition 70. Thus, the probability that the rounded solution does not satisfy Q is at most

$$
O\left(\sqrt{\epsilon} + \sqrt{(\kappa + 2c) \log \frac{1}{\epsilon}} + \frac{c}{\gamma} \right).
$$

This completes the proof of Lemma 74 and Theorem 73.

6.4.2 Warm-up for AT: Oblivious LP rounding algorithm for OR

As a stepping-stone for our algorithm for AT presented in the next section, we present a robust algorithm for the OR polymorphism. Horn-SAT is an example of a (P)CSP for which OR is a polymorphism. A robust algorithm for Horn-SAT was found previously by Zwick [Zwi98]. (See also the matching hardness result by Guruswami and Zhou[GZ12].) We now present an algorithm for PCSPs with the OR polymorphism achieving similar guarantees. As mentioned earlier, besides a polymorphic generalization of the Horn-SAT robust algorithm, our motivation is a warm-up for the algorithm for AT polymorphism in the next section.

As the OR operator is naturally over the $0/1$ basis, we shall assume that the predicates $P, Q \subseteq \{0,1\}^k$ for this section.

1. Solve the basic LP and obtain the value y_i for each variable $i \in [n]$.
2. Let T be a geometric progression with first term $2\sqrt{\epsilon}$, last term $1/(2k)$ and spacing between terms is at least k, where k is an upper bound on the maximum clause size of Γ.
3. Sample a uniformly random threshold $t \in T$.
4. For each $i \in [n]$, round as follows

$$x_i = \begin{cases} 1 & y_i \geq t \\ 0 & \text{otherwise.} \end{cases}$$

Theorem 76. *Let* Γ *be a PCSP such that* $\mathrm{OR} \in \mathrm{Pol}(\Gamma)$*. Let* Ψ *be an instance of* $\mathrm{PCSP}(\Gamma)$ *for which there is a basic LP solution with a completeness of* $1-\epsilon$*. Then, with high probability our algorithm finds an integral assignment to* Ψ *which satisfies* $1 - O_\Gamma(1/\log(1/\epsilon))$ *fraction of the clauses in expectation.*

Proof. Since we assume k is a constant, the size of T is $O_\Gamma(1/\log(1/\epsilon))$. As with the analysis of MAJ, we fix a single clause (P, Q) and analyze that. In fact, by Markov's inequality we may assume that $1-\sqrt{\epsilon}$ fraction of the clauses have value at least $1-\sqrt{\epsilon}$.

Since $\mathrm{OR} \in \mathrm{Pol}(P, Q)$, we in fact have that $\mathrm{OR} \in \mathrm{Pol}(Q)$ as well (see [BG21b]). Thus, by Schaefer's theorem [Sch78] $\mathrm{CSP}(Q)$ and thus $\mathrm{PCSP}(P, Q)$ can be ppp-reduced to k-Horn-SAT. Thus, we assume we are working with k-CNF clauses with at least $k-1$ variables negated.

Consider a Horn-SAT clause on variables $x_1, \ldots, x_k$ with value at least $1-\sqrt{\epsilon}$. First assume none of the variables are negated. Then, we have that $y_1 + y_2 + \cdots + y_k \geq 1 - \sqrt{\epsilon}$. By pigeonhole, we must have some $y_i \geq (1-\sqrt{\epsilon})/k \geq 1/(2k)$. Thus, $y_i \geq t$ for all $t \in T$. So the clause must be satisfied.

Otherwise, without loss of generality assume that x_1 is negated. Then, we have that $(1-y_1) + y_2 + \cdots + y_k \geq 1 - \sqrt{\epsilon}$. In particular, there exists $i \in \{2, 3, \ldots, k\}$ such that $y_i \geq (y_1 - \sqrt{\epsilon})/(k-1)$. If $y_1 \leq 2\sqrt{\epsilon}$, then $x_1 = 0$ with certainty. Otherwise, $y_i \geq y_1/(2(k-1))$, so there exists at most one $t \in T$ which would round y_i to 0 but y_1 to 1. Therefore, the probability the clause is satisfied is at least $1 - 1/|T| = 1 - 1/\log(1/\epsilon)$. This completes the proof. ∎

6.4.3 Algorithm for AT

We now show how to combine ideas for the MAJ and OR algorithms to give an algorithm for AT, Suppose that $\mathrm{Pol}(\Gamma)$ contains AT_L for every odd integer L. We state our algorithm below.

1. Solve the basic SDP and obtain vectors $\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_n \in \mathbb{R}^n$.
2. Sample a vector $\zeta \in \mathbb{R}^n$ by choosing each coordinate independently from $\mathcal{N}(0,1)$.
3. Choose δ uniformly at random from $\{p, rp, \ldots, r^\kappa p\}$ where $p = \epsilon^{0.24}$, and $r = \kappa = \Theta\left(\frac{\log \frac{1}{\epsilon}}{\log\log \frac{1}{\epsilon}}\right)$ such that $r^\kappa p = \epsilon^{0.20}$.
4. For every $i \in [n]$, let $\mathbf{v}_i = \alpha_i \mathbf{v}_0 + \mathbf{v}'_i$, where $\mathbf{v}'_i$ is orthogonal to $\mathbf{v}_0$. We set x_i as follows.

$$x_i = \begin{cases} -1, \text{ if } \langle \zeta, \mathbf{v}'_i \rangle \geq \delta \alpha_i \left| \langle \zeta, \mathbf{v}_0 \rangle \right|. \\ +1, \text{ otherwise.} \end{cases}$$

We show that the above algorithm is a robust algorithm for every PCSP with AT polymorphisms.

Theorem 77. *Let Γ be a Boolean PCSP such that $\mathrm{AT} \in \mathrm{Pol}(\Gamma)$. Let Ψ be an instance of $\mathrm{PCSP}(\Gamma)$ for which there is a basic SDP solution with completeness at least $1-\epsilon$. Then, the integeral solution output by the above algorithm satisfies at least $1 - O_\Gamma\left(\frac{\log \frac{1}{\epsilon}}{\log\log \frac{1}{\epsilon}}\right)$ fraction of constraints of Ψ.*

Theorem 77 together with Theorem 73 completes the proof of Theorem 63.

For ease of notation, we just use $O()$ instead of $O_\Gamma()$ when Γ is clear from the context. Consider an arbitrary predicate pair $\{P, Q\}$ with arity k i.e., $P \subseteq Q \subseteq \{-1,+1\}^k$. As (P,Q) contains AT of all odd arities as polymorphisms, $O_{AT}(P) \subseteq Q$. Henceforth, in our analysis, we assume that $Q = O_{AT}(P)$. Furthermore, by using Markov's inequality, at least $1-\sqrt{\epsilon}$ fraction of the constraints have SDP error at most $\sqrt{\epsilon}$. We restrict our analysis to these constraints with SDP error $c \leq \sqrt{\epsilon}$ using and and show that the rounded solution satisfies the predicate Q with probability at least $1 - O\left(\frac{\log\log \frac{1}{\epsilon}}{\log \frac{1}{\epsilon}}\right)$.

We study the case when P and Q are symmetric in Section 6.4.3 and handle the general case in Section 6.4.4. We use the fact that P, Q are symmetric to reduce to a special case. Note that if $Q = \{-1,+1\}^k$, we are done. Henceforth, we assume that $Q \neq \{-1,+1\}^k$. Using the properties of symmetric PCSPs with AT polymorphisms proved by Brakensiek and Guruswami [BG21b], we get the following:

Lemma 78. *Suppose that (P,Q) is a symmetric Boolean PCSP with arity k such that AT_L is a polymorphism of (P,Q) for every odd integer L, and $P \subseteq Q \subseteq \{-1,+1\}^k$, $Q \neq \{-1,+1\}^k$. Then, either of the two following conditions hold:*

1. *$P = Q$ and $P \subseteq \{(-1,-1,\ldots,-1),(+1,+1,\ldots,+1)\}$.*
2. *There exists $l \in \{1,2,\ldots,k-1\}$ such that $P = \mathrm{Ham}_k\{l\}$, and $Q = \mathrm{Ham}_k\{1,2,\ldots,k-1\}$.*

Proof. Using Claim 4.6 in [BG21b], if $\mathrm{Ham}_k\{l_1,l_2\} \subseteq P$ where $l_1 \neq l_2$, $\{l_1,l_2\} \neq \{0,k\}$, we get that $Q = \{-1,+1\}^k$, a contradiction. Thus, either $P = \mathrm{Ham}_k\{l\}$ for some $l \in \{0,1,\ldots,k\}$

or $P = \mathrm{Ham}_k\{0, k\}$. If $P \subseteq \mathrm{Ham}_k\{0, k\}$, we get that $P = Q$. If not, then $P = \mathrm{Ham}_k\{l\}$ for $l \in \{1, 2, \ldots, k-1\}$, and by the same Claim 4.6 in [BG21b], we get that $Q = O_{AT}(P) = \mathrm{Ham}_k\{1, 2, \ldots, k-1\}$. □

We consider the case when $P = Q$ and $P \subseteq \mathrm{Ham}_k\{0, k\}$.

Lemma 79. *For every constraint (P, Q) where $P = Q$, $P \subseteq \mathrm{Ham}_k\{0, k\}$ such that the basic SDP has error at most $\sqrt{\epsilon}$ on a constraint using (P, Q), the above algorithm succeeds with probability $1 - O\left(\frac{\log\log\frac{1}{\epsilon}}{\log\frac{1}{\epsilon}}\right)$.*

We defer the proof of Lemma 79 to Section 6.7.

For the rest of the section, we assume that there exists $l \in \{1, 2, \ldots, k-1\}$ such that $P = \mathrm{Ham}_k\{l\}$, and $Q = \mathrm{NAE}_k$. More generally, we assume that there exist $\mathbf{w} \in \mathbb{R}^k, b \in \mathbb{R}$ such that $w_i > 0 \forall i \in [k]$, $\sum_i w_i > |b|$, and $\mathbf{w} \cdot a = b$ for all $a \in P$. Here, for the case when $P = \mathrm{Ham}_k\{l\}$, we can take $\mathbf{w} = (1, 1, \ldots, 1)$ and $b = 2l - k$.

First, we prove some properties of the basic SDP vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$ corresponding to the variables used in the constraint. As is the case with the MAJ algorithm, we have $K = 2^k$ probabilities $p_1, p_2, \ldots, p_K$ corresponding to the assignments $a_1, a_2, \ldots, a_K \in \{-1, +1\}^k$ such that

$$\sum_{i \in K, a_i \in P} p_i \geq 1 - \sqrt{\epsilon}$$

We use the basic SDP properties to get the following. Let $\mathbf{v}_s = \sum_{i \in [k]} w_i \mathbf{v}_i$, and let $\mathbf{v}_s = \alpha \mathbf{v}_0 + \mathbf{v}'_s$, where $\langle \mathbf{v}_0, \mathbf{v}'_s \rangle = 0$.

1. (First moments). We have

$$\alpha = \sum_{i \in [k]} w_i \alpha_i = \sum_{i \in [k]} w_i \langle \mathbf{v}_i, \mathbf{v}_0 \rangle = \sum_{i \in [K]} p_i \mathbf{w} \cdot a_i = b + \kappa$$

   where $|\kappa| = O(\sqrt{\epsilon})$.

2. (Second moments). We have

$$\|\mathbf{v}_s\|_2^2 = \sum_{i,j \in [k]} w_i w_j \langle \mathbf{v}_i, \mathbf{v}_j \rangle = \sum_{i \in [K]} p_i (a_i \cdot \mathbf{w})^2 = b^2 + \kappa'$$

   where $|\kappa'| = O(\sqrt{\epsilon})$.

Thus, we get $\|\mathbf{v}'_s\|_2^2 = \|\mathbf{v}_s\|_2^2 - \alpha^2 = (b^2 + \kappa') - (b + \kappa)^2$ which is at most $O(\sqrt{\epsilon})$.

We are now ready to analyze the algorithm. We consider two cases separately:

Case 1. Suppose that there exists $i \in [k]$ such that $\|\mathbf{v}'_i\|_2 \geq k\delta r^2$. We claim that in this case, the rounded solution satisfies Q with probability at least $1 - O(\frac{1}{r})$.

Note that $\langle \zeta, \mathbf{v}'_j \rangle \sim \mathcal{N}(0, \left\|\mathbf{v}'_j\right\|_2^2)$ for every $j \in [k]$. Suppose that we have $\left\|\mathbf{v}'_j\right\| \geq \delta r^2$ for some $j \in [k]$. Using Proposition 70, this implies that $|\langle \zeta, \mathbf{v}'_j \rangle| \geq r\delta$ with probability at least $1 - \frac{1}{r}$. Furthermore, as $\langle \zeta, \mathbf{v}_0 \rangle \sim \mathcal{N}(0, 1)$, using Proposition 71, we get that $|\langle \zeta, \mathbf{v}_0 \rangle| \leq r$ with probability at least $1 - O(\epsilon)$. Thus, with probability at least $1 - O(\frac{1}{r})$, x_j is set to be equal to $+1$ if $\langle \zeta, \mathbf{v}'_j \rangle > 0$, and -1 otherwise.

Hence, in order to show that the rounded solution satisfies Q, it suffices to show that there exist $i_1, i_2 \in [k]$ such that $|\langle \zeta, \mathbf{v}'_{i_1}\rangle| \geq \delta r$, $|\langle \zeta, \mathbf{v}'_{i_2}\rangle| \geq \delta r$, and $\langle \zeta, \mathbf{v}'_{i_1}\rangle$ and $\langle \zeta, \mathbf{v}'_{i_2}\rangle$ have opposite signs. As $\|\mathbf{v}'_i\|_2 \geq k\delta r^2$, with probability at least $1 - O(\frac{1}{r})$, we have that $|\langle \zeta, \mathbf{v}'_i\rangle| \geq k\delta r$. Recall that $\|\mathbf{v}'_s\|_2 \leq \epsilon^{0.25} \leq \delta$. Thus, $|\langle \zeta, \mathbf{v}'_s\rangle| \leq r\delta$ with probability at least $O(\frac{1}{r})$. As $|\langle \zeta, \mathbf{v}'_i\rangle| \geq k\delta r$, there exists $i' \in [k], i' \neq i$ such that $|\langle \zeta, \mathbf{v}'_{i'}\rangle| \geq k\delta r$, and $\langle \zeta, \mathbf{v}'_i\rangle$ and $\langle \zeta, \mathbf{v}'_{i'}\rangle$ have opposite signs. Thus, with probability at least $1 - O(\frac{1}{r})$, i and i' are rounded to different values, which implies that the rounded solution satisfies Q.

Case 2**.** Suppose that for every $i \in [k]$, we have $\|\mathbf{v}'_i\|_2 \leq \frac{\delta}{2r^2}$.

As $\langle \zeta, \mathbf{v}'_i\rangle \sim \mathcal{N}(0, \|\mathbf{v}'_i\|_2^2)$, using Proposition 71, we get that with probability at least $1 - O(\frac{1}{r})$, for every $i \in [k]$, $|\langle \zeta, \mathbf{v}'_i\rangle| \leq \frac{\delta}{2r}$. On the other hand, using Proposition 70, we have that $|\langle \zeta, \mathbf{v}_0\rangle| \geq \frac{1}{r}$ with probability at least $1 - \frac{1}{r}$. Furthermore, As $\alpha_i^2 + \|\mathbf{v}'_i\|_2^2 = 1$ for every $i \in [k]$, we get that $|\alpha_i| \geq 1 - \delta \geq \frac{1}{2}$ for every $i \in [k]$. Thus, with probability at least $1 - O(\frac{1}{r})$, for every $i \in [k]$, x_i is set to be $+1$ if $\alpha_i \leq 0$, and -1 otherwise. Combining this with the fact that $\sum_i w_i\alpha_i = b + O(\sqrt{\epsilon})$, and that $\sum_i w_i > b$ and $\sum_i w_i > -b$, for small enough ϵ, we get the rounded solution has variables assigned $+1$ and -1.

Completing the proof. We finish the proof by showing that with probability at least $1 - O(\frac{1}{r})$, at least one of the above two cases hold. None of the above two cases hold if for some $i \in [k]$, we have

$$\frac{\delta}{2r^2} < \|\mathbf{v}'_i\|_2 < k\delta r^2$$

Or equivalently,

$$\frac{\|\mathbf{v}'_i\|_2}{kr^2} < \delta < \|\mathbf{v}'_i\|_2\, 2r^2$$

This holds with probability at most $O(\frac{1}{r})$ for every value of $\|\mathbf{v}'_i\|$ as we are picking δ from $\{p, rp, \ldots, r^\kappa p\}$ uniformly at random.

6.4.4 General case for AT

For a vector $\mathbf{w} \in \mathbb{R}^k$, define $\operatorname{sgn}(\mathbf{w})_i$ to be -1 if $w_i \leq 0$ and $+1$ otherwise. Define Γ_{AT} to be the following family of *weighted hyperplane predicates*:

$$\begin{aligned}\Gamma_{AT} := \{&(P_{\mathbf{w},b} := \{x \in \{-1,+1\}^k : \mathbf{w} \cdot x = b\},\\ &Q_{\mathbf{w},b} := \{-1,+1\}^k \setminus \{\operatorname{sgn}(\mathbf{w}), -\operatorname{sgn}(\mathbf{w})\}) : b \in \mathbb{Q}, \mathbf{w} \in (\mathbb{Q} \setminus \{0\})^k,\\ &\mathbf{w}.\operatorname{sgn}(\mathbf{w}) > b, -\mathbf{w}.\operatorname{sgn}(\mathbf{w}) < b\}\end{aligned}$$

We observe that these predicates indeed have AT of all odd arities as polymorphisms.

Claim 80. $\mathrm{AT} \in \mathrm{Pol}(\Gamma_{AT})$

Proof. Fix $b \in \mathbb{Q}$ and $\mathbf{w} \in (\mathbb{Q} \setminus \{0\})^k$. Let $(P_{w,b}, Q_{w,b})$ be the corresponding predicate for these values. It sufficies to show that $\mathrm{AT} \in \mathrm{Pol}(P_{w,b}, Q_{w,b})$. Fix an odd arity L and pick $x_1, \ldots, x_L \in P_{w,b}$. Observe that

$$\mathrm{AT}(x_1, \ldots, x_L) = \operatorname{sgn}(x_1 - x_2 + \cdots + x_L).$$

Further, $\mathbf{w} \cdot (x_1 - x_2 + \cdots + x_L) = b$. This implies that $\operatorname{sgn}(x_1 - x_2 + \cdots + x_L) \neq \operatorname{sgn}(\mathbf{w})$ as otherwise,

$$b = \mathbf{w} \cdot (x_1 - x_2 + \cdots + x_L) \geq \mathbf{w} \cdot \operatorname{sgn}(\mathbf{w}) > b.$$

where we used the fact that the absolute value of each entry in $x_1 - x_2 + \ldots + x_L$ is at least 1. By a similar argument, $\operatorname{sgn}(x_1 - x_2 + \cdots + x_L) \neq -\operatorname{sgn}(\mathbf{w})$. Thus, $\mathrm{AT}(x_1, \ldots, x_L) \in Q_{w,b}$, as desired. □

Note that our algorithm in the previous subsection gives a robust algorithm for these predicates as well. Henceforth, we reduce arbitrary Boolean PCSPs containing AT of all odd arities to these predicates via ppp reductions that preserve robust algorithms.

Let Γ_{const} be the PCSP where constants can be specified. That is $\{(\{-1\}, \{-1\}), (\{+1\}, \{+1\})\}$.

Theorem 81. *Let Γ be any PCSP for which $\mathrm{AT} \in \mathrm{Pol}(\Gamma)$. Then, there is a ppp-reduction from Γ to $\Gamma_{\mathrm{AT}} \cup \Gamma_{const}$.*

Note that the analysis in Section 6.4.3 shows that our algorithm is a robust algorithm for Γ_{AT}. Furthermore, Lemma 79 shows that our algorithm is a robust algorithm for Γ_{const} as well. Together with Theorem 81, we get that our algorithm is a robust algorithm for every PCSP that contains AT of all odd arities as polymorphisms.

To prove Theorem 81, we need to use the following lemma implicit in [BG21b]. We present the proof in Section 6.7 for the sake of completeness.

Lemma 82. *Let P be a predicate such that there is non-trivial dependence in each coordinate (i.e., for each x_i, there exist assignements with $x_i = -1$ and $x_i = +1$). Then, $O_{\mathrm{AT}}(P) = \{\operatorname{sgn}(x - y) : x, y \in \mathrm{Aff}(P), \forall i, x_i \neq y_i\}$, where $\mathrm{Aff}(P)$ is the affine hull of P.*

Proof of Theorem 81. Fix a pair of predicates $(P, Q) \in \Gamma$. It sufficies to show that there is a PPP-reduction from (P, Q) to $\Gamma_{\mathrm{AT}} \cup \Gamma_{const}$. If P has coordinates of fixed value, we can use a gadget reduction from Γ_{const} to simulate these values. Thus, we assume that P has non-trivial dependence in each coordinate, and thus we apply Lemma 82 to get that $Q \supseteq O_{\mathrm{AT}}(P) = \{\operatorname{sgn}(x - y) : x, y \in \mathrm{Aff}(P), \forall i, x_i \neq y_i\}$. We may without loss of generality assume that $Q = \{\operatorname{sgn}(x - y) : x, y \in \mathrm{Aff}(P), \forall i, x_i \neq y_i\}$.

For every $\mathbf{x} \in \{-1, +1\}^k \setminus Q$, we find $\mathbf{w}, b$ such that $(P_{\mathbf{w},b} := \{x \in \{-1, +1\}^k : \mathbf{w} \cdot x = b\}, Q_{\mathbf{w},b} := \{-1, +1\}^k \setminus \{\operatorname{sgn}(\mathbf{w}), -\operatorname{sgn}(\mathbf{w})\})$ satisfy $P \subseteq P_{\mathbf{w},b}$ and $\operatorname{sgn}(\mathbf{w}) = \mathbf{x}$. By applying this for every $\mathbf{x} \in \{-1, +1\} \setminus Q$, we get a set of predicate pairs $(P_1, Q_1), (P_2, Q_2), \ldots, (P_L, Q_L)$ with $L \leq 2^k$ such that

1. $P \subseteq P_i$ for every $i \in [L]$.
2. $(P_i, Q_i) \in \Gamma_{AT}$ for every $i \in [L]$.
3. $\bigcap_{i \in [L]} Q_i = Q$.

This directly gives a ppp-reduction from (P, Q) to Γ_{AT}.

Henceforth, our goal is to show that for every $\mathbf{x} \in \{-1, +1\}^k \setminus Q$, we can find $\mathbf{w}, b$ such that $(P_{\mathbf{w},b} := \{x \in \{-1, +1\}^k : \mathbf{w} \cdot x = b\}$ satisfies $P \subseteq P_{\mathbf{w},b}$, $\mathbf{w} \cdot \operatorname{sgn}(\mathbf{w}) > b, \mathbf{w} \cdot \operatorname{sgn}(\mathbf{w}) > -b$ and $\operatorname{sgn}(\mathbf{w}) = \mathbf{x}$. Without loss of generality, we can assume that $\mathbf{x} = (+1, +1, \ldots, +1)$. Fix an arbitrary vector $\overline{\mathbf{x}} \in P$ such that $\overline{\mathbf{x}} \notin \{(-1, -1, \ldots, -1), (+1, +1, \ldots, +1)\}$. Such a vector is guaranteed to exist as P does not contain $\overline{x}$ and has non-trivial dependence on each coordinate. Let H be a subspace of $\mathbb{R}^k$ defined as follows:

$$H := \{\mathbf{y} - \overline{\mathbf{x}} : \mathbf{y} \in \mathrm{Aff}(P)\}$$

As $\mathbf{x} \notin O_{AT}(P)$, using Lemma 82, we get that for every $\mathbf{z} \in H$, $\mathrm{sgn}(\mathbf{z}) \neq \mathbf{x}$, or in other words, there is no $\mathbf{z} \in H$ with $z_i > 0$ for all $i \in [k]$. Using Claim 83, we can obtain $\mathbf{w}$ such that $\mathbf{w} \cdot \mathbf{y} = 0$ for all $\mathbf{y} \in H$, and $w_i > 0$ for all $i \in [k]$. This shows that $\mathbf{w} \cdot \mathbf{y} = b$ for every $\mathbf{y} \in P$, where $b = \mathbf{w} \cdot \overline{\mathbf{x}}$ satisfies $\sum_i w_i > b$, $\sum_i w_i > -b$.

□

Claim 83. *Let H be a subspace of $\mathbb{R}^k$ such that there is no $\mathbf{y} \in H$ with $y_i > 0$ for all i. Then, there exists $\mathbf{w}$ with $w_i > 0$ for all i and $\mathbf{w} \cdot \mathbf{y} = 0$ for all $\mathbf{y} \in H$.*

Proof. Since H and the positive orthant are both convex bodies, there exists $\mathbf{v} \in \mathbb{R}^k$ and $b \in \mathbb{R}$ such that for all $\mathbf{w}$ in the positive orthant, $\mathbf{w} \cdot \mathbf{v} > b$ and for all $\mathbf{y} \in H$, $\mathbf{y} \cdot \mathbf{v} \leq b$. Taking the limit as $\mathbf{w} \to 0$, we have that $b \leq 0$. Further, since H is a subspace passing through the origin, we must have that $b = 0$ and that $\mathbf{y} \cdot \mathbf{v} = 0$ for all $\mathbf{y} \in H$. Thus, $\mathbf{v}$ is normal to H. Note that $\mathbf{v}$ has all coordinates positive as $\mathbf{v} \cdot \mathbf{w} > 0$ for all $\mathbf{w}$ in the positive orthant. □

6.5 Unique Games based Hardness

In this section, we prove Theorem 64.

First, we use the analysis of AT and MAJ polymorphisms for symmetric PCSPs with folding and idempotence in [BG21b] to show that we can relax Γ into one of five candidate PCSP types.

Lemma 84. *Let $\Gamma = (P, Q)$ be a Boolean symmetric folded idempotent PCSP such that $\mathrm{MAJ}_{L_1}, \mathrm{AT}_{L_2} \notin \mathrm{Pol}(\Gamma)$ for some odd integers L_1, L_2. Then, there exists a PCSP $\Gamma' = (P, Q)$ that is a relaxation of Γ that is equal to either of the following:*

1. *k is even, and $\Gamma_1 = (P, Q), P = \mathrm{Ham}_k\{\frac{k}{2}\}, Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{b\}$ where $b \in \{1, k-1\}$.*
2. *k is odd, $\Gamma_2 = (P, Q), P = \mathrm{Ham}_k\{l, \frac{k+1}{2}\}$, $Q = \mathrm{Ham}_k\{0, 1, 2, \ldots, k-1\}$, where $l \leq \frac{k-1}{2}$.*
3. *$\Gamma_3 = (P, Q), P = \mathrm{Ham}_k\{l, k\}$, $Q = \mathrm{Ham}_k\{1, 2, \ldots, k\}$, where $l \neq 0, l \leq \frac{k-1}{2}$.*
4. *$\Gamma_4 = (P, Q), P = \mathrm{Ham}_k\{l\}, Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{0, k-1\}$ where $l \in \{1, 2, \ldots, k-1\}, l \leq \frac{k-1}{2}$.*
5. *$\Gamma_5 = (P, Q), P = \mathrm{Ham}_k\{1, k\}, Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{b\}$ for arbitrary $b \in \{0, 1, \ldots, k\}$.*

We defer the proof of Lemma 84 to Section 6.7.

We show that the PCSPs Γ_1, $\Gamma_2, \Gamma_3, \Gamma_4$ and Γ_5 do not have robust algorithms by showing integrality gaps for the basic SDP relaxation of them. Raghavendra's result for CSPs [Rag08] shows that integrality gaps for the basic SDP relaxation can be translated to Unique Games Conjecture(UGC) [Kho02a] based inapproximability results. In fact, his result is verbatim applicable to Promise CSPs as well.

Theorem 85 (Special case of [Rag08] for PCSPs when the completeness value of SDP is 1**).** *Suppose that for a Promise CSP Γ, there is a finite integrality gap for the basic SDP i.e., there is a finite instance I of Γ where basic SDP error is zero but I is not satisfiable by Γ, even using the weak form of the constraints. Then, there exists a constant $s < 1$ that is a function of Γ, I such that for every $\epsilon > 0$, assuming the Unique Games Conjecture, given an instance of Γ, there is no polynomial time algorithm to distinguish between the two cases:*

1. *(Completeness) There exists an assignment that satisfies $1 - \epsilon$ fraction of the strong constraints.*
2. *(Soundness) No assignment satisfies s fraction of the weak constraints.*

To obtain integrality gaps for the basic SDP relaxation for a PCSP, we study colorings of the n dimensional sphere $\mathbb{S}^n$ that satisfies certain properties. First, we define certain notations that we need. For a predicate $P \subseteq \{-1, +1\}^k$, we say that a set of vectors $V = \{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_k\}$ are a P-configuration with respect to another vector $\mathbf{v}_0$ if the Gram matrix of $\mathbf{v}_0, \mathbf{v}_1, \dots, \mathbf{v}_k$ is in $\mathrm{KZ}(P)$. Fix a vector $\mathbf{v}_0$ and we say that a coloring $f : \mathbb{S}^n \to \{-1, +1\}$ respects the PCSP (P, Q) if for every P-configuration $V = \{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_k\}$ with respect to $\mathbf{v}_0$, we have that the colors of the vectors satisfy Q, i.e.,

$$(f(\mathbf{v}_1), f(\mathbf{v}_2), \dots, f(\mathbf{v}_k)) \in Q$$

More generally, we say that a coloring $f : \mathbb{S}^n \to D_2$ respects a PCSP Γ over a pair of domains D_1, D_2 if it respects every predicate pair in Γ.

We show that the absence of such sphere coloring respecting Γ for some finite n gives an integrality gap for basic SDP relaxation of Γ.

Lemma 86. *For every PCSP Γ over a pair of domains (D_1, D_2), the Basic SDP solves $\mathrm{PCSP}(\Gamma)$ if and only if for every $n \geq 1$, there exists a coloring $f : \mathbb{S}^n \to D_2$ that respects Γ.*

Proof. Via a compactness[12] argument (e.g., like the De Brujin-Erdos theorem [BE51], for more details see Remark 7.13 of [Bar+21] or [CŽ22a]), we can infer that there is a coloring $f : \mathbb{S}^n \to D_2$ respecting Γ if and only if for every finite subset $S \subseteq \mathbb{S}^n$, there exists a coloring $f_S : S \to D_2$ that respects Γ.

First, assume that the Basic SDP solves $\mathrm{PCSP}(\Gamma)$. For any finite subinstsance $S \subset \mathbb{S}^n$, we construct an instance I of Γ where we add a constraint over P_i using the variables $x_{\mathbf{v}_1}, \dots, x_{\mathbf{v}_{r_i}}$ corresponding to the vectors $V = (\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{r_i})$ if V is a P_i-configuration. We have that $x_{\mathbf{v}} \mapsto \mathbf{v}$ is a valid SDP solution. Thus, there exists an assignment to the variables that satisfies I, or equivalently, there exists $f_S : S \to D_2$ that respects the PCSP Γ. And thus, there exists a coloring $f : \mathbb{S}^n \to \{-1, +1\}$ that respects Γ.

Second, fix an integer $n \geq 1$ and assume that there exists a coloring $f : \mathbb{S}^n \to D_2$ that respects Γ. We seek to show that the Basic SDP solves $\mathrm{PCSP}(\Gamma)$. Take an arbitrary instance I of $\mathrm{PCSP}(\Gamma)$ such that there is a solution to Basic SDP with objective value zero. Thus, there exists a dimension n and a mapping $x \mapsto \mathbf{v}_x$ of the variables to n-dimensional SDP vectors. By setting x to $f(\mathbf{v}_x)$ for each variable x, we get an assignment satisfying all the constraints in I. Thus, the Basic SDP solves Γ. $\square$

[12] We assume the axiom of choice.

Theorem 85 together with Lemma 86 shows that if a PCSP Γ over a pair of domains (D_1, D_2) does not admit a sphere coloring $f : \mathbb{S}^n \to D_2$ that respects Γ for some positive integer n, then, Γ does not admit a robust algorithm, assuming the Unique Games Conjecture. Thus, our goal is to show that the PCSPs mentioned in Lemma 84 do not admit sphere coloring that respects them. While we fail to achieve this for Γ_1 and Γ_4, for the rest of the PCSPs, we show the absence of sphere coloring, which in turn implies Unique Games based hardness of obtaining robust algorithms.

In the rest of this section, we first prove a lemma regarding sphere Ramsey theory that we will use later. Then, we show that the earlier mentioned PCSPs do not have folded sphere coloring respecting them, thus showing that they don't admit robust algorithms. We remark that as we restrict ourselves to Boolean folded PCSPs, we only study the colorings $f : \mathbb{S}^n \to \{-1, +1\}$ that are folded i.e., $f(-\mathbf{x}) = -f(\mathbf{x})$.

6.5.1 Sphere Ramsey Theory

For a finite set $S \subseteq \mathbb{R}^{n+1}$, we use $\rho(S)$ to denote the sphere of the smallest radius that contains S as a subset. Matoušek and Rödl [MR95] proved the following:

Theorem 87. *Let S be the set of vertices of a simplex such that $\rho(S) < 1$. Then, for every positive integer $r \geq 2$, there exists $n_0 := n_0(S, r)$ such that for every $n \geq n_0$, for every partition $f : \mathbb{S}^n \to [r]$, there exists $S' \subseteq \mathbb{S}^n$ that is monochromatic and is congruent to S.*

We will use this to show the following lemma regarding sphere colorings.

Lemma 88. *Fix an integer $k \geq 3$ and $r \geq 2$. There exists $n_0 := n_0(k)$ such that for every $n \geq n_0$ and coloring $f : \mathbb{S}^n \to [r]$ and $\gamma \in \mathbb{R}$ with $\frac{-1}{k-1} < \gamma < 1$, there exists a monochromatic set of vectors $V = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_k\} \subseteq \mathbb{S}^n$ such that $\boldsymbol{v}_i \cdot \boldsymbol{v}_j = \gamma$ for every $i \neq j$.*

Proof. Consider an arbitrary set $S = \{\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_k\}$ of k unit vectors in $\mathbb{S}^n$ such that $\mathbf{u}_i \cdot \mathbf{u}_j = \gamma$ for every $i \neq j$. Such a set S is guaranteed to exist when n is large enough. We show that the vectors are affinely independent: suppose for contradiction that there exists reals $c_1, c_2, \ldots, c_k$ not all zero, $\sum_i c_i = 0$ and $\sum_i c_i \mathbf{u}_i = 0$. We have

$$0 = \mathbf{u}_1 \cdot \left(\sum_i c_i \mathbf{u}_i \right) = c_1 + \gamma(c_2 + \ldots + c_k) = c_1 + \gamma(-c_1)$$

implying that $c_1 = 0$. The same argument shows that $c_i = 0$ for all $i \in [k]$, a contradiction.

As S is affinely independent, they can be viewed as vertices of a simplex. Furthermore, the set of vectors can be embedded on a sphere of radius strictly smaller than 1: let $\alpha \in \mathbb{R}$ such that $0 < \alpha < \frac{2}{k}$, and let $\mathbf{u}_s = \sum_{i \in [k]} \mathbf{u}_i$, $\mathbf{c} = \alpha \mathbf{u}_s$. We have

$$\|\mathbf{u}_s\|_2^2 = \sum_i \|\mathbf{u}_i\|_2^2 + 2 \sum_{i \neq j} \mathbf{u}_i \cdot \mathbf{u}_j = k + \frac{k(k-1)}{\gamma}$$

Note that

$$\begin{aligned}\|\mathbf{u}_i - \mathbf{c}\|_2^2 &= \|\mathbf{u}_i\|_2^2 + \|c\|_2^2 - 2\mathbf{c}\cdot\mathbf{u}_i \\ &= 1 + \alpha^2(k + \frac{k(k-1)}{\gamma}) - 2\alpha(1+(k-1)\gamma) \\ &= 1 - k(1+(k-1)\gamma)\alpha\left(\alpha - \frac{2}{k}\right)\end{aligned}$$

which is strictly smaller than 1 when $0 < \alpha < \frac{2}{k}$. Thus, all the vectors are on a sphere centered at $\mathbf{c}$ and radius strictly smaller than 1, implying that $\rho(S) < 1$. Now, we can use Theorem 87 on S and f to obtain the required set of vectors V. $\square$

While Theorem 87 is applicable to a wide range of sets S, we sometimes need to apply it to sets S that do not form a simplex or have $\rho(S) = 1$. Towards this, we use the "Spreads" based idea in [MR95] to obtain a version of Theorem 87 directly for certain sets S where Theorem 87 is not applicable.

We use the following notion of Spread vectors from [MR95]. For an integer n, a vector $\mathbf{a} \in \mathbb{R}^k$, and a set $J \subseteq [n]$ of cardinality k with $J = \{j_1, j_2, \ldots, j_k\}$, we let

$$\mathrm{Spread}_n(\mathbf{a}, J) = \sum_{i=1}^{k} a_i e_{j_i}$$

where $e_1, e_2, \ldots, e_n$ is an orthonormal basis of $\mathbb{R}^n$. For a set $I \subseteq [n]$, we let

$$\mathrm{Spread}_n(\mathbf{a}, I) = \{\mathrm{Spread}_n(\mathbf{a}, J) : J \subseteq I, |J| = k\}$$

We get the following as a direct application of the hypergraph Ramsey theorem.

Lemma 89. *([MR95]) For every $\boldsymbol{a} \in \mathbb{R}^k, n, k$, there exists N such that in any coloring f : $\mathrm{Spread}_N(\boldsymbol{a}, [N]) \to [r]$, there exists I with $|I| = n$ such that $\mathrm{Spread}_N(\boldsymbol{a}, I)$ is monochromatic with respect to f, i.e., $\exists p \in [r]$ such that $f(\boldsymbol{v}) = p$ for all $\boldsymbol{v} \in \mathrm{Spread}_N(\boldsymbol{a}, I)$.*

Lemma 89 implies the following immediately.

Corollary 90. *Suppose that $U = \{\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_k\}$ be a set of k unit vectors such that $\boldsymbol{u}_i \in \mathrm{Spread}_N(\boldsymbol{a}, [N])\ \forall i \in [k]$ for an integer N, and a vector $\boldsymbol{a} \in \mathbb{R}^N$ with $\|\boldsymbol{a}\|_2 = 1$. Then there exists $n_0 := n_0(U, \boldsymbol{a}, N)$ such that for every $n \geq n_0, r$, for every sphere coloring $f : \mathbb{S}^n \to [r]$, there exists a set of k vectors $V = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k\}$ that are all colored the same, and $\boldsymbol{v}_i \cdot \boldsymbol{v}_j = \boldsymbol{u}_i \cdot \boldsymbol{u}_j$ for every $i, j \in [k]$.*

We use Corollary 90 to obtain a couple of lemmas regarding sphere colorings. For ease of notation, we call a set of k unit vectors $V = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k\}$ to be *k-regular* if $\mathbf{v}_i \cdot \mathbf{v}_j = -\frac{1}{k-1}$ for every $i \neq j$.

Lemma 91. *Fix an integer $k \geq 2$. There exists $n_0 := n_0(k)$ such that for every $n \geq n_0$ and folded coloring $f : \mathbb{S}^n \to \{-1, +1\}$, there exist a k-regular set of vectors $V = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_k\} \subseteq \mathbb{S}^n$ such that exactly $k-1$ vectors in V are colored -1.*

Proof. We construct a set of k unit vectors $V = \{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_k\}$ in $\mathrm{Spread}_N(\mathbf{a}, [N])$ such that $\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{k-1}, -\mathbf{v}_k\}$ is a k-regular set, where $N, \mathbf{a}$ depend only on k, and $\|\mathbf{a}\|_2 = 1$. Using Corollary 90, we can infer that in the coloring f, there exist k vectors $V = \{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_k\}$ that are all assigned the same color, such that $\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{k-1}, -\mathbf{v}_k\}$ is a k-regular set. As f is folded, this implies that there is a k-regular set in which exactly $k-1$ vectors are assigned the color -1.

Thus our goal is to construct k unit vectors $V = \{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_k\}$ in $\mathrm{Spread}_N(\mathbf{a}, [N])$ such that $\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{k-1}, -\mathbf{v}_k\}$ is a k-regular set. Or equivalently, we construct the vectors $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{k-1} \in \mathrm{Spread}_N(\mathbf{a}, [N])$ and $\mathbf{v}_k \in \mathrm{Spread}_N(-\mathbf{a}, [N])$ such that $\{\mathbf{v}_1, \dots, \mathbf{v}_k\}$ is a k-regular set. We set $\gamma = \frac{1}{\sqrt{2(k-1)}}$ and $\mathbf{a} = (\gamma, -\gamma, \gamma, -\gamma, \dots, -\gamma) \in \mathbb{R}^{2(k-1)}$. We set $\mathbf{v}_i = \mathrm{Spread}_n(\mathbf{a}, J_i), i \in [k-1], \mathbf{v}_k = \mathrm{Spread}(-\mathbf{a}, J_k)$ where $J_1, J_2, \dots, J_k$ such that $|J_i| = 2(k-1)$ for every $i \in [k]$. We obtain these sets by induction on k. First, we consider the base case when $k = 2$. In this case, we set $J_1 = J_2 = \{1, 2\}$ and $N = 2$ suffices. The vectors are the following:

$$\mathbf{v}_1 = (\gamma, -\gamma)$$
$$\mathbf{v}_2 = (-\gamma, \gamma)$$

where $\gamma = \frac{1}{\sqrt{2}}$. Note that the above two vectors are a 2-regular set, and $\mathbf{v}_1 \in \mathrm{Spread}_2(\mathbf{a}, [2]), \mathbf{v}_2 \in \mathrm{Spread}_2(-\mathbf{a}, [2])$ with $\mathbf{a} = (\gamma, -\gamma)$. Now, suppose that $J_1, J_2, \dots, J_k, N$ are such that $\mathbf{v}_i = \mathrm{Spread}_N(\mathbf{a}, J_i), i \in [k-1], \mathbf{v}_k = \mathrm{Spread}_N(-\mathbf{a}, J_k)$ satisfy the property that $\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_k\}$ is a k-regular set with $\mathbf{a} = (\gamma, -\gamma, \dots, \gamma, -\gamma) \in \mathbb{R}^{2(k-1)}, \gamma = \frac{1}{\sqrt{2(k-1)}}$. We construct $J'_1, J'_2, \dots, J'_{k+1}$ such that $\mathbf{v}'_i = \mathrm{Spread}_{N'}(\mathbf{a}', J'_i), i \in [k], \mathbf{v}'_{k+1} = \mathrm{Spread}_{N'}(-\mathbf{a}', J_{k+1})$ satisfy the property that $\{\mathbf{v}'_1, \mathbf{v}'_2, \dots, \mathbf{v}'_{k+1}\}$ is a $(k+1)$-regular set with $\mathbf{a}' = (\gamma', -\gamma', \dots, \gamma', -\gamma') \in \mathbb{R}^{2k}, \gamma' = \frac{1}{\sqrt{2k}}$.

1. For every $i \in [k-1]$, we obtain J'_i from J_i by adding two new elements.

   $$J'_i = J_i \cup \{N + 2i, N + 2i + 1\}$$

   This ensures that $\mathbf{v}'_i \cdot \mathbf{v}'_j = -(\gamma')^2$ for every $i, j \in [k-1], i \neq j$.
2. We obtain J_{k+1} from J_k by adding two new elements.

   $$J_{k+1} = J_k \cup \{N + 1, N + 2k\}$$

   This ensures that $\mathbf{v}'_i \cdot \mathbf{v}'_{k+1} = -(\gamma')^2$ for every $i \in [k-1]$.
3. Finally, we set J_k.

   $$J_k = \{N + 1, N + 2, \dots, N + 2k\}$$

   This ensures that $\mathbf{v}'_i \cdot \mathbf{v}'_k = -(\gamma')^2$ for every $i \in [k+1], i \neq k$.

We illustrate our construction by obtaining the vectors for the case when $k = 3$ and $k = 4$:

$$\begin{array}{llll}
\mathbf{v}_1 = (& \alpha, -\alpha, & 0, \alpha, & -\alpha, 0) \\
\mathbf{v}_2 = (& 0, 0, & \alpha, -\alpha, & \alpha, -\alpha) \\
\mathbf{v}_3 = (& -\alpha, \alpha, & -\alpha, 0, & 0, \alpha)
\end{array}$$

$$
\begin{array}{lllllll}
\mathbf{v}_1 = (& \beta, -\beta, & 0,\beta, & -\beta,0, & 0,\beta, & -\beta,0, & 0,0) \\
\mathbf{v}_2 = (& 0,0, & \beta, -\beta, & \beta, -\beta, & 0,0, & 0,\beta, & -\beta,0) \\
\mathbf{v}_3 = (& 0,0, & 0,0, & 0,0, & \beta, -\beta, & \beta, -\beta, & \beta, -\beta) \\
\mathbf{v}_4 = (& -\beta,\beta, & -\beta,0, & 0,\beta, & -\beta,0, & 0,0, & 0,\beta)
\end{array}
$$

where $\alpha = \frac{1}{2}$ and $\beta = \frac{1}{\sqrt{6}}$.

As the pairwise inner product of every pair in $\{\mathbf{v}'_1, \mathbf{v}'_2, \ldots, \mathbf{v}'_{k+1}\}$ is equal to $-(\gamma')^2 = -\frac{1}{k}$, we get that these set of vectors are a $(k+1)$-regular set, completing the inductive proof.

□

6.5.2 Absence of sphere coloring

First, we show the absence of sphere coloring respecting Γ_1 using Lemma 91.

Lemma 92. *Fix an even integer $k \geq 4$. There exists an integer n_0 such that for every $n \geq n_0$, there is no folded $f : \mathbb{S}^n \to \{-1, +1\}$ that respects $\Gamma_1 = (P, Q), P = Ham_k\{\frac{k}{2}\}, Q = Ham_k\{0, 1, \ldots, k\} \setminus \{b\}$ where $b \in \{1, k-1\}$.*

Proof. Suppose for contradiction that such a folded f exists. Fix a vector $\mathbf{v}_0 \in \mathbb{S}^n$. We get the P-configuration of vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$ such that the gram matrix of $\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_k$ is a uniform convex combination from the vertices of $\mathrm{KZ}(P)$. The vectors satisfy this if we have

1. (First moments.) $\mathbf{v}_i \cdot \mathbf{v}_0 = 0$ for every $i \in [k]$.
2. (Second moments.) $\mathbf{v}_i \cdot \mathbf{v}_j = \frac{2\binom{\frac{k}{2}}{2} - \frac{k^2}{4}}{\binom{k}{2}} = \frac{-1}{k-1}$.

We restrict ourselves to vectors orthogonal to $\mathbf{v}_0$, and apply Lemma 91 to obtain a set of k vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$ such that $\mathbf{v}_i \cdot \mathbf{v}_j = \frac{-1}{k-1}$ and exactly $k-1$ of $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k\}$ are colored -1. By negating these vectors if needed, we get a set of k vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$ such that $\mathbf{v}_i \cdot \mathbf{v}_j = \frac{-1}{k-1}$ and exactly b of $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k\}$ are colored $+1$. Thus, there exists a P-configuration of vectors whose f value contains exactly b $+1$s, a contradiction. □

We show the absence of sphere coloring respecting Γ_2, Γ_3 and Γ_4 using Lemma 88.

Lemma 93. *Fix an odd integer $k \geq 3$ and integer $l : 0 \leq l \leq \frac{k-1}{2}$. There exists an integer n_0 such that for every $n \geq n_0$, there is no folded $f : \mathbb{S}^n \to \{-1, +1\}$ that respects $\Gamma_2 = (P, Q), P = Ham_k\{l, \frac{k+1}{2}\}$, $Q = Ham_k\{0, 1, 2, \ldots, k-1\}$.*

Proof. Fix $\mathbf{v}_0 \in \mathbb{S}^n$. The P-configuration that we consider is a set of vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$ such that the gram matrix of $\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_k$ is obtained by first sampling $i \in \{l, \frac{k+1}{2}\}$ where we set

$$
i = \begin{cases} l \text{ with probability} \frac{1}{1-s} \\ \frac{k+1}{2} \text{ with probability} \frac{-s}{1-s} \end{cases}
$$

where $s = l - (k - l) < 0$. Then, we sample a uniformly random vertex from $\mathrm{KZ}(\mathrm{Ham}_k\{i\})$. We obtain the following properties:

1. (First moments). $\mathbf{v}_i \cdot \mathbf{v}_0 = 0$ for every $i \in [k]$.
2. (Second moments). $\mathbf{v}_i \cdot \mathbf{v}_j = \gamma$ for every $i \neq j$. Furthermore, we get that $\frac{-1}{k} \leq \gamma < 1$.

Now, restricting ourselves to the vectors in $\mathbb{S}^n$ that are orthogonal to $\mathbf{v}_0$, and using Theorem 87, we get that there exists a P-configuration of vectors that are all colored the same. By taking the negation of these vectors if needed, we get our required claim. $\square$

Lemma 94. *Fix integers k, l such that $0 < l \leq \frac{k-1}{2}$. Then, there exists an integer n_0 such that for every $n \geq n_0$, there is no folded $f : \mathbb{S}^n \to \{-1, +1\}$ that respects $\Gamma_3 = (P, Q), P = Ham_k\{l, k\}$, $Q = Ham_k\{1, 2, \ldots, k\}$, where $l \neq 0, l \leq \frac{k-1}{2}$.*

Proof. Fix $\mathbf{v}_0 \in \mathbb{S}^n$. We pick the P-configuration along the same lines as in Lemma 93. We sample $i \in \{l, k\}$ with

$$i = \begin{cases} l \text{ with probability } \frac{k}{k-s} \\ k \text{ with probability } \frac{-s}{k-1} \end{cases}$$

where $s = l - (k - l) < 0$. As before, we sample a uniformly random vertex from $\mathrm{KZ}(\mathrm{Ham}_k\{i\})$. We get

1. (First moments). $\mathbf{v}_i \cdot \mathbf{v}_0 = \left(\frac{k}{k-s}\right) \frac{s}{k} + \left(\frac{-s}{k-s}\right) 1 = 0$ for every $i \in [k]$.
2. (Second moments). For every $i \neq j \in [k]$, we get

$$\mathbf{v}_i \cdot \mathbf{v}_j = \left(\frac{k}{k-s}\right) \frac{\binom{l}{2} + \binom{k-l}{2} - l(k-l)}{\binom{k}{2}} + \left(\frac{-s}{k-s}\right) 1$$

The function $\binom{l}{2} + \binom{k-l}{2} - l(k-l)$ is a decreasing function when $l \in [1, \frac{k-1}{2}]$. When $l = \frac{k-1}{2}$, the value of it is equal to $\frac{-1}{k}$. Thus, we get that for every $1 \leq l \leq \frac{k-1}{2}$, we get that $\binom{l}{2} + \binom{k-l}{2} - l(k-l) \geq \frac{-1}{k}$. Thus, we get that

$$\mathbf{v}_i \cdot \mathbf{v}_j = \gamma$$

for every $i \neq j$, and $\frac{-1}{k} < \gamma < 1$. We restrict ourselves to vectors in $\mathbb{S}^n$ that are orthogonal to $\mathbf{v}_0$, and applying Theorem 87, we get that for any coloring $f : \mathbb{S}^n \to \{-1, +1\}$, there is a monochromatic P-configuration that we described. By negating the vectors if needed, we get our required proof. $\square$

Lemma 95. *Fix integers $k \geq 3, l \in \{1, \ldots, k-1\}, l \leq \frac{k-1}{2}$. There exists integer n_0 such that for every $n \geq n_0$, there does not exist coloring $f : \mathbb{S}^n \to \{0, 1\}$ that is folded i.e., $f(-x) = -f(x)$, and respects the PCSP $\Gamma_4 = (P, Q), P = \mathrm{Ham}_k\{l\}, Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{0, k-1\}$.*

Proof. We partition the predicate P into P_1 and P_{-1} depending on the value of the first element, i.e.,

$$P_i = \{\mathbf{x} \in P : x_1 = i\}, i \in \{-1, +1\}$$

We pick the P-configuration as follows: sample i from $\{-1, +1\}$ uniformly at random, then, sample a uniformly random vertex from $\mathrm{KZ}(P_i)$. We get

1. (First moments). By our choice of P_is, we get that

$$\mathbf{v}_1 \cdot \mathbf{v}_0 = 0$$

By using symmetry of the rest of the variables, we get that

$$\mathbf{v}_i \cdot \mathbf{v}_0 = \frac{2l-k}{k-1} \forall i \in \{2,3,\ldots,k\}.$$

For ease of notation, let $\alpha = \frac{2l-k}{k-1}$.

2. (Second moments). We get

$$\mathbf{v}_1 \cdot \mathbf{v}_i = \frac{-1}{k-1} \, \forall i \in \{2,3,\ldots,k\}.$$
$$\mathbf{v}_i \cdot \mathbf{v}_j = \frac{(2l-k)^2 - (k-2)}{(k-1)(k-2)} \, \forall i,j \in \{2,3,\ldots,k\}, i \neq j$$

For ease of notation, let $\beta = \frac{(2l-k)^2-(k-2)}{(k-1)(k-2)}$.

We collect the following fact for later use:

$$\begin{aligned}
\left\| \sum_{i=1}^{k} \mathbf{v}_i \right\|^2 &= k + 2 \sum_{i,j \in [k], i \neq j} \mathbf{v}_i \cdot \mathbf{v}_j \\
&= k + 2 \sum_{i \in \{2,\ldots,k\}} \mathbf{v}_1 \cdot \mathbf{v}_i + 2 \sum_{i,j \in \{2,\ldots,k\}, i \neq j} \mathbf{v}_i \cdot \mathbf{v}_j \\
&= k + 2(k-1)\frac{-1}{k-1} + (k-1)(k-2)\frac{(2l-k)^2-(k-2)}{(k-1)(k-2)} \\
&= (2l-k)^2.
\end{aligned}$$

Thus, we get

$$\left\| \sum_{i=1}^{k} \mathbf{v}_i \right\| = k - 2l \tag{6.3}$$

Our goal is to show that there exists n_0 such that for every $n \geq n_0$, for every folded sphere coloring $f : \mathbb{S}^n \to \{-1,+1\}$ and $\mathbf{v}_0 \in \mathbb{S}^n$, there exists a set of k vectors $V = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k\}$ that satisfy the above first and second moments, and exactly b vectors in V are colored $+1$, where $b \in \{0, k-1\}$. For $i \in [k]$, let $\mathbf{v}'_i$ be the component of $\mathbf{v}_i$ orthogonal to $\mathbf{v}_0$:

$$\mathbf{v}'_i = \mathbf{v}_i - (\mathbf{v}_i \cdot \mathbf{v}_0)\mathbf{v}_0, i \in [k]$$

Note that $\|\mathbf{v}'_i\| = \sqrt{1-\alpha^2}$ for $i \in \{2,\ldots,k\}$ and $\|\mathbf{v}'_1\| = 1$. We let $\mathbf{u}_i = \frac{\mathbf{v}'_i}{\|\mathbf{v}'_i\|}$. We have

$$\begin{aligned}
\mathbf{u}_i \cdot \mathbf{u}_j &= \frac{\mathbf{v}'_i \cdot \mathbf{v}'_j}{1-\alpha^2} \\
&= \frac{(\mathbf{v}_i - \alpha\mathbf{v}_0) \cdot (\mathbf{v}_j - \alpha\mathbf{v}_0)}{1-\alpha^2} \\
&= \frac{\beta - \alpha^2}{1-\alpha^2} \, \forall i,j \in \{2,\ldots,k\}, i \neq j.
\end{aligned}$$

For ease of notation, $\gamma = \frac{\beta-\alpha^2}{1-\alpha^2}$.

Using Equation (6.3), we get the following bound on γ that we will use later.

We have

$$\begin{aligned} k - 2l &= \left\| \sum_{i=1}^{k} \mathbf{v}_i \right\| \\ &= \left\| \mathbf{v}_1 + \sum_{i=2}^{k} \mathbf{v}'_i + \alpha(k-1)\mathbf{v}_0 \right\| \\ &= \left\| \mathbf{u}_1 + \sqrt{1-\alpha^2} \sum_{i=2}^{k} \mathbf{u}_i + (2l-k)\mathbf{v}_0 \right\| \end{aligned}$$

As $\mathbf{u}_i \cdot \mathbf{v}_0 = 0$ for every $i \in [k]$, we get that

$$\mathbf{u}_1 + \sqrt{1-\alpha^2} \sum_{i=2}^{k} \mathbf{u}_i = 0$$

Thus,

$$\left\| \sum_{i=2}^{k} \mathbf{u}_i \right\|^2 = \frac{1}{1-\alpha^2}$$

On the other hand,

$$\begin{aligned} \left\| \sum_{i=2}^{k} \mathbf{u}_i \right\|^2 &= k - 1 + 2 \sum_{i,j \in \{2,\ldots,k\}, i \neq j} \mathbf{u}_i \cdot \mathbf{u}_j \\ &= k - 1 + (k-1)(k-2)\gamma \end{aligned}$$

Thus, we get

$$\gamma = \frac{1}{(1-\alpha^2)(k-1)(k-2)} - \frac{1}{k-2} \tag{6.4}$$

We apply Lemma 88 on the following coloring of the sphere. For a vector $\mathbf{u} \in \mathbb{S}^n$ such that $\mathbf{u} \cdot \mathbf{v}_0 = 0$, let $f' : \mathbb{S}^{n-1} \to \{-1, +1\}^2$ be defined as

$$f'(\mathbf{u}) = \left(f\left(\alpha\mathbf{v}_0 + \sqrt{1-\alpha^2}\mathbf{u} \right), f\left(\alpha\mathbf{v}_0 - \sqrt{1-\alpha^2}\mathbf{u} \right) \right)$$

Using Lemma 88 on f' combined with the fact that $\gamma > \frac{-1}{k-2}$ obtained from Equation (6.4), we can infer that there exist $k-1$ unit vectors $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_{k-1} \in \mathbb{S}^n$ such that $\mathbf{u}_i \cdot \mathbf{v}_0 = 0$ for all i, $\mathbf{u}_i \cdot \mathbf{u}_j = \gamma$ for all $i \neq j$ and $f'(\mathbf{u}_i) = f'(\mathbf{u}_j)$ for all $i \neq j, i, j \in [k-1]$.

We define $\mathbf{v}_1^{(1)}, \mathbf{v}_2^{(1)}, \ldots, \mathbf{v}_k^{(1)}, \mathbf{v}_1^{(2)}, \ldots, \mathbf{v}_k^{(2)}$ as follows. For $i \in \{2, 3, \ldots, k\}$, we let

$$\begin{aligned} \mathbf{v}_i^{(1)} &= \alpha\mathbf{v}_0 + \sqrt{1-\alpha^2}\mathbf{u}_{i-1} \\ \mathbf{v}_i^{(2)} &= \alpha\mathbf{v}_0 - \sqrt{1-\alpha^2}\mathbf{u}_{i-1} \end{aligned}$$

We let

$$\mathbf{v}_1^{(1)} = -\frac{\sum_{i=1}^{k-1} \mathbf{u}_i}{\left\|\sum_{i=1}^{k-1} \mathbf{u}_i\right\|}$$

and $\mathbf{v}_1^{(2)} = -\mathbf{v}_1^{(1)}$. We now prove that the set of vectors $\mathbf{v}_1^{(1)}, \mathbf{v}_2^{(1)}, \dots, \mathbf{v}_k^{(1)}$ and the set of vectors $\mathbf{v}_1^{(2)}, \mathbf{v}_2^{(2)}, \dots, \mathbf{v}_k^{(2)}$ are a P-configuration with first and second moments as computed earlier, where we sampled i from $\{-1, +1\}$ uniformly at random and sampled a uniformly random vertex from $\mathrm{KZ}(P_i)$.

1. (First moments). As $\mathbf{u}_i \cdot \mathbf{v}_0 = 0$ for all $i \in [k-1]$, we get that

$$\mathbf{v}_1^{(1)} \cdot \mathbf{v}_0 = \mathbf{v}_1^{(2)} \cdot \mathbf{v}_0 = 0$$

   and

$$\mathbf{v}_i^{(1)} \cdot \mathbf{v}_0 = \mathbf{v}_i^{(2)} \cdot \mathbf{v}_0 = \alpha \ \ \forall i \in \{2, \dots, k\}$$

2. (Second moments). We have

$$\begin{aligned}
\mathbf{v}_1^{(1)} \cdot \mathbf{v}_i^{(1)} &= -\frac{(\sum_{j=1}^{k-1} \mathbf{u}_j) \cdot (\alpha \mathbf{v}_0 + \sqrt{1-\alpha^2}\mathbf{u}_{i-1})}{\left\|\sum_{j=1}^{k-1} \mathbf{u}_j\right\|} \\
&= \frac{\sqrt{1-\alpha^2}}{k-1} - \frac{(\sum_{j=1}^{k-1} \mathbf{u}_j) \cdot (\sum_{j=1}^{k-1} \mathbf{u}_j)}{\left\|\sum_{j=1}^{k-1} \mathbf{u}_j\right\|} \\
&= -\frac{\sqrt{1-\alpha^2}}{k-1} \left\|\sum_{j=1}^{k-1} \mathbf{u}_j\right\| \\
&= -\frac{\sqrt{1-\alpha^2}}{k-1} \sqrt{k-1+2\frac{(k-1)(k-2)}{2}\gamma} \\
&= -\frac{\sqrt{1-\alpha^2}}{k-1} \sqrt{\frac{1}{1-\alpha^2}} \text{ Using } Equation \text{ (6.4)} \\
&= \frac{-1}{k-1} \ \forall i \in \{2, 3, \dots, k\}.
\end{aligned}$$

   Furthermore,

$$\begin{aligned}
\mathbf{v}_i^{(1)} \cdot \mathbf{v}_j^{(1)} &= (\alpha \mathbf{v}_0 + \sqrt{1-\alpha^2}\mathbf{v}_i) \cdot (\alpha \mathbf{v}_0 + \sqrt{1-\alpha^2}\mathbf{v}_j) \\
&= \alpha^2 + (1-\alpha^2)\gamma \\
&= \beta \ \forall i, j \in \{2, 3, \dots, k\}, i \neq j
\end{aligned}$$

   Similarly, we have

$$\begin{aligned}
\mathbf{v}_1^{(2)} \cdot \mathbf{v}_i^{(2)} &= \frac{(\sum_{j=1}^{k-1} \mathbf{u}_j) \cdot (\alpha \mathbf{v}_0 - \sqrt{1-\alpha^2}\mathbf{u}_{i-1})}{\left\|\sum_{j=1}^{k-1} \mathbf{u}_j\right\|} = \frac{-1}{k-1} \ \forall i \in \{2, 3, \dots, k\}. \\
\mathbf{v}_i^{(2)} \cdot \mathbf{v}_j^{(2)} &= (\alpha \mathbf{v}_0 - \sqrt{1-\alpha^2}\mathbf{v}_i) \cdot (\alpha \mathbf{v}_0 - \sqrt{1-\alpha^2}\mathbf{v}_j) \\
&= \alpha^2 + (1-\alpha^2)\gamma = \alpha^2 + (1-\alpha^2)\gamma = \beta \ \forall i, j \in \{2, 3, \dots, k\}, i \neq j
\end{aligned}$$

Thus, the set of vectors $\mathbf{v}_1^{(1)}, \mathbf{v}_2^{(1)}, \ldots, \mathbf{v}_k^{(1)}$ and the set of vectors $\mathbf{v}_1^{(2)}, \mathbf{v}_2^{(2)}, \ldots, \mathbf{v}_k^{(2)}$ are a P-configuration. As $f'(\mathbf{u}_1) = f'(\mathbf{u}_2) = \ldots = f'(\mathbf{u}_{k-1})$, we can infer that $f(\mathbf{v}_2^{(1)}) = f(\mathbf{v}_3^{(1)}) = \ldots = f(\mathbf{v}_k^{(1)})$ and $f(\mathbf{v}_2^{(2)}) = f(\mathbf{v}_3^{(2)}) = \ldots = f(\mathbf{v}_k^{(2)})$. Furthermore, as $\mathbf{v}_1^{(1)} = -\mathbf{v}_1^{(2)}$ and f is folded, we can infer that $f(\mathbf{v}_1^{(1)}) = -f(\mathbf{v}_1^{(2)}$. Thus, there exists $p \in \{1, 2\}$ such that $f(\mathbf{v}_1^{(p)}) = -1$. Thus, there are either 0 or $k-1$ vectors among $\mathbf{v}_1^{(p)}, \mathbf{v}_2^{(p)}, \ldots, \mathbf{p}_k^{(1)}$ that are colored $+1$ according to f, contradicting the fact that f respects the PCSP (P, Q). $\square$

Finally, we show the absence of sphere coloring for Γ_5.

Lemma 96. *Fix integers $k \geq 3, b \in \{0, 1, \ldots, k\} \setminus \{1, k\}$. There exists integer n_0 such that for every $n \geq n_0$, there does not exist coloring $f : \mathbb{S}^n \to \{0, 1\}$ that is folded i.e., $f(-x) = -f(x)$, and respects the PCSP $\Gamma_5 = (P, Q), P = Ham_k\{1, k\}, Q = Ham_k\{0, 1, \ldots, k\} \setminus \{b\}$.*

We dedicate the rest of the section to proving Lemma 96. We pick the configuration of vectors along the same lines as in Lemma 93. Fix $\mathbf{v}_0 \in \mathbb{S}^n$. The P-configuration that we study is a set of vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$ that is obtained by first sampling $i \in \{1, k\}$ such that

$$i = \begin{cases} 1 \text{ with probability} \frac{k}{2k-2} \\ k \text{ with probability} \frac{k-2}{2k-2} \end{cases}$$

Then, we sample a uniform point from $\mathrm{KZ}(\mathrm{Ham}_k\{i\})$. We get the following properties:

1. (First moments). $\mathbf{v}_i \cdot \mathbf{v}_0 = \left(\frac{k}{2k-2}\right) \frac{2-k}{k} + \left(\frac{k-2}{2k-2}\right) 1 = 0$ for every $i \in [k]$.
2. (Second moments). For every $i \neq j \in [k]$, we get

$$\mathbf{v}_i \cdot \mathbf{v}_j = \left(\frac{k}{2k-2}\right) \frac{\binom{k-1}{2} - (k-1)}{\binom{k}{2}} + \left(\frac{k-2}{2k-2}\right) 1 = \frac{k-3}{k-1}$$

For ease of notation, let $\alpha = \frac{k-3}{k-1}$. Furthermore, by restricting ourselves to vectors in $\mathbb{S}^n$ that are orthogonal to $\mathbf{v}_0$, we just focus on P-configurations that are a set of k unit vectors all of whose pairwise inner product is equal to α. We refer to these set of vectors i.e., a set V of k unit vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k \in \mathbb{S}^n$ an α-configuration if the inner product of every pair of them is equal to α. Given the folded sphere coloring f, our goal is to show that there is an α-configuration of vectors V among which exactly b of them are assigned $+1$.

Unlike the earlier studied PCSPs, here, the setting when $b = 0$ is relatively straightforward, simply because $\alpha \geq 0$. $\alpha \geq 0$ implies that there are an arbitrarily large number of unit vectors (as we can pick n to be large enough) all of whose pairwise inner product is equal to α. In particular, we pick a set of $2k - 1$ unit vectors all of whose pairwise inner product is equal to α. Among those, k of them are colored the same according to f. By taking the negation of these if needed, we can infer that there are α-configurations that are all colored $+1$, and also α-configurations that are all colored -1.

Before delving further, we handle the case when $\alpha = 0$ i.e., when $k = 3$. In this case, we just pick a set of k unit vectors that are all orthogonal to each other and their negations. Note that these are $2k$ pairwise orthogonal vectors where exactly k of them are colored $+1$ according to f.

Thus, we can pick k pairwise orthogonal vectors from this set where exactly b of them are colored $+1$ according to f. Henceforth, we assume that $\alpha > 0$.

To show that there are α-configurations that have exactly b vectors that are colored $+1$, we show a connectivity lemma (Lemma 99) where we prove that between any two α-configurations, there exists a path using $O_{k,\alpha}(1)$ α-configurations where we change a single vector at each step in the path. As there is an α-configuration where all are k vectors are colored $+1$, and the α-configuration obtained by negating these vectors where all the vectors are colored -1, the connectivity lemma then shows that for every $b \in \{0, 1, \ldots, k\}$, there exists an α-configuration that has exactly b vectors that are colored $+1$.

We first prove the following simplified version of the connectivity lemma that we use to prove Lemma 99.

Lemma 97. *Given an α-configuration $U = \{\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_k\} \subseteq \mathbb{S}^n$, and a unit vector $\boldsymbol{w} \in \mathbb{S}^n$ that is orthogonal to each vector in U, there exists $L := L(k, \alpha)$ and a set of α-configurations $V_1, V_2, \ldots, V_L$ such that*

1. *The consecutive configurations differ in a single vector i.e.,$|V_i \cap V_{i+1}| = k - 1$ for every $i \in [L-1]$.*
2. *Final configuration contains $\boldsymbol{w}$ i.e., $\boldsymbol{w} \in V_L$, and the initial configuration V_1 is equal to U.*

Proof. We prove the lemma by studying the inner product of w with an α-configuration V, which is equal to all zeroes initially when $V = U$, and changing V one vector at a time such that the inner product of V with w eventually reaches all αs. Towards this end, for an α-configuration V, we define the matrix $(k+1) \times (k+1)$ matrix $I(V, w) = [\mathbf{v}_1 \mathbf{v}_2 \ldots \mathbf{v}_k \mathbf{w}]^T [\mathbf{v}_1 \mathbf{v}_2 \ldots \mathbf{v}_k \mathbf{w}]$ defined as follows:

$$I(V, \mathbf{w})_{i,j} = \begin{cases} \langle \mathbf{v}_i, \mathbf{v}_j \rangle \text{ if } 1 \leq i, j \leq k. \\ \langle \mathbf{v}_i, \mathbf{w} \rangle \text{ if } i = k+1, 1 \leq j \leq k. \\ \langle \mathbf{w}, \mathbf{v}_j \rangle \text{ if } j = k+1, 1 \leq i \leq k. \\ \langle \mathbf{w}, \mathbf{w} \rangle = 1, \text{ if } i = j = k+1. \end{cases}$$

Starting with $I(V, \mathbf{w})$ where $V = U$, our goal is to change one vector in V at a time so that we eventually reach a configuration where the last column in $I(V, \mathbf{w})$ is equal to $(\alpha, \alpha, \ldots, \alpha, 1)$. Note that changing one vector in V corresponds to changing a single value in the last column (and the corresponding value in the last row) in $I(V, \mathbf{w})$. We show that the opposite direction also holds i.e., by changing a single value in the last column (and the corresponding value in the last row) of $I(V, \mathbf{w})$, we obtain a new matrix that is $I(V', \mathbf{w})$ with V' being different from V only in a single vector, as long as the new matrix is positive semidefinite.

Claim 98. *Suppose that A is a $m \times m$ real symmetric positive semidefinite matrix with $A = U^T U$ with $U = [\boldsymbol{u}_1 \boldsymbol{u}_2 \ldots \boldsymbol{u}_m]$ where $\boldsymbol{u}_i \in \mathbb{R}^n$ with $n \geq m$, and A' is another real symmetric positive semidefinite matrix such that A' and A differ only in $A_{1,2} = A_{2,1}$. Then, there exists $U' = [\boldsymbol{u}'_1 \boldsymbol{u}_2 \ldots \boldsymbol{u}_m]$ such that $A' = (U')^T U'$.*

Proof. As A' is a positive semidefinite matrix, there exist $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m \in \mathbb{R}^n$ such that $A' = V^T V$ where $V = [\mathbf{v}_1 \mathbf{v}_2 \ldots \mathbf{v}_m]$. Let $A[2:m]$ is a $(m-1) \times (m-1)$ submatrix of A excluding the first row and the column i.e., $A[2:m]_{i,j} = A_{i+1,j+1}$. We define the corresponding submatrix of A' as $A'[2:m]$. Note that $A[2:m] = A'[2:m]$. Let $U[2:m] = [\mathbf{u}_2\mathbf{u}_3\dots\mathbf{u}_m]$, and similarly, let $V[2:m] = [\mathbf{v}_2\mathbf{v}_3\dots\mathbf{v}_m]$. Note that $A[2:m] = U[2:m]^T U[2:m]$, and $A'[2:m] = V[2:m]^T V[2:m]$. However, as $A[2:m] = A'[2:m]$, we can infer that there exists a unitary $n \times n$ matrix H such that $U[2:m] = HV[2:m]$. Now, setting $U' = HV$, we get the required matrix U' with $A' = (U')^T U'$. $\square$

Thus, our goal is to obtain a series of $(k+1)\times(k+1)$ real symmetric positive semidefinite matrices $M_1, M_2, \dots, M_L$ such that

1. $M_1 = I(U, \mathbf{w})$.
2. The diagonal entries of M_i for every $i \in [L]$ are all equal to 1.
3. All the off diagonal entries of M_L are equal to α.
4. For every $i \in [L-1]$, M_i and M_{i+1} differ only in one element in the last column (and the corresponding element in the last row).

Towards this end, for $\epsilon \geq 0$, $0 \leq \gamma \leq \alpha$, and $d \in [k]$, we define the $(k+1)\times(k+1)$ matrix $M(\gamma, \epsilon, d)$ as follows:

$$M(\gamma,\epsilon,d)_{i,j} = \begin{cases} 1, \text{ if } i = j. \\ \alpha, \text{ if } 1 \leq i,j \leq k. \\ \gamma+\epsilon, \text{ if } i = k+1, 1 \leq j \leq d \text{ or } j = k+1, 1 \leq i \leq d. \\ \gamma, \text{ if } i = k+1, d+1 \leq j \leq k \text{ or } j = k+1, d+1 \leq i \leq k. \end{cases}$$

Note that $I_{U,\mathbf{w}} = M(0,0,k)$, and our goal M_L is equal to $M(\alpha, 0, k)$. We define the sequence of positive semidefinite matrices $M(0,0,k), M(0,\epsilon,1), M(0,\epsilon,2), \dots, M(0,\epsilon,k), M(\epsilon,\epsilon,1), M(\epsilon,\epsilon,2), \dots, M($ $\epsilon,\epsilon,k)$. Note that at each step, we change a single element in the last column (and the corresponding element in the last row).

The final step is to show that when we set $\epsilon \leq \frac{1-\alpha}{k}$, $M(\gamma,\epsilon,d)$ is positive semidefinite for every $d \in [k], 0 \leq \gamma \leq \alpha$. This follows from a simple calculation.

$$\begin{aligned} x^T M(\gamma,\epsilon,d)x &= \gamma(\sum_{i=1}^{k+1} x_i)^2 + (\alpha-\gamma)(\sum_{i=1}^{k} x_i)^2 + (1-\alpha)\sum_{i=1}^{k} x_i^2 + (1-\gamma)x_{k+1}^2 + \epsilon(x_{k+1})(\sum_{i=1}^{d} x_i) \\ &\geq (1-\alpha)\sum_{i=1}^{k+1} x_i^2 + \epsilon(x_{k+1})(\sum_{i=1}^{d} x_i) \\ &\geq \left(\frac{1-\alpha}{k}\right)\left(k\sum_{i=1}^{k+1} x_i^2 - \left|(x_{k+1})(\sum_{i=1}^{d} x_i)\right|\right) \geq 0\,. \end{aligned}$$ $\square$

Now, we prove the connectivity lemma.

Lemma 99. *Fix an integer $k \geq 2$ and $0 < \alpha < 1$. Suppose that U and V are two α-configurations in $\mathbb{S}^n$. Then, there exists $n_0 := n_0(k,\alpha)$, and $L := L(k,\alpha)$ such that as long as $n \geq n_0$, there exist α-configurations $V_1, V_2, \dots, V_L$ such that*

1. *The end points are U and V i.e., $U = V_1, V = V_L$.*
2. *Any two consecutive configurations differ in exactly one vector i.e., $|V_i \cap V_{i+1}| = k - 1$ for every $i \in [L-1]$.*

Proof. We use induction on k. First, we consider the case when $k = 2$. Let $U = \{\mathbf{u}_1, \mathbf{u}_2\}$, and $V = \{\mathbf{v}_1, \mathbf{v}_2\}$ be two α-configurations. Consider an arbitrary vector $\mathbf{w}$ that is orthogonal to all the vectors in U and V. Such a $\mathbf{w}$ is guaranteed to exist when n is large enough. Now, using Lemma 97, we can infer that there exists a configuration $W = \{\mathbf{w}, \mathbf{w}'\}$ such that there is a path of length $O_\alpha(1)$ from U to W, and from V to W. Thus, there exists a path of length $O_\alpha(1)$ from U to V.

Assume that the proof holds for $k-1$, and we are given the configurations U and V consisting of k vectors each. We choose a vector $\mathbf{w}$ that is orthogonal to each of the vectors in U and V. Using Lemma 97, there are configurations $X = \{\mathbf{w}, \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{k-1}\}$ and $Y = \{\mathbf{w}, \mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{k-1}\}$ such that there is a path of length $O_{\alpha,k}(1)$ from U to X and from V to Y. Now, our goal is to show that there is a path from X to Y of length $O_{\alpha,k}(1)$. We achieve this by restricting ourselves to components orthogonal to $\mathbf{w}$ of the $(k-1)$-sized configurations $X' = \{\mathbf{x}'_1, \mathbf{x}'_2, \ldots, \mathbf{x}'_{k-1}\}$ and $Y' = \{\mathbf{y}'_1, \mathbf{y}'_2, \ldots, \mathbf{y}'_{k-1}\}$, where $\mathbf{x}'_i = \mathbf{x}_i - \langle \mathbf{x}_i, \mathbf{w}\rangle \mathbf{w}$ for each $i \in [k-1]$ (and similarly for $\mathbf{y}'_i$. Note that X' and Y' are $(\alpha - \alpha^2)$ configurations, and by the induction hypothesis, there exists a path from X' to Y' using only vectors orthogonal to $\mathbf{w}$. Adding the component along $\mathbf{w}$, we get a path from X to Y of length $O_{\alpha,k}(1)$, finishing the proof. ☐

We are now ready to prove Lemma 96.

Proof. Suppose for contradiction that there exists a coloring $f : \mathbb{S}^n \to \{0, 1\}$ that is folded and respects the PCSP Γ_5. Consider an arbitrary set of vectors $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{2k-1}\}$ that are all orthogonal to $\mathbf{v}_0$, and have pairwise inner product α. Such a set is guaranteed to exist as $\alpha \geq 0$. There exists a set of k vectors among these that are all assigned the same color in f. Let these form the configuration U, and the set of negations of these vectors be the configuration V. Using Lemma 99, there exists a path from U to V where we change a single vector in each step. Note that the endpoints of the path have 0 and k vectors assigned $+1$ respectively. Since we change at most one vector at a time, there exists a configuration where we have exactly b 1s, a contradiction. ☐

Lemma 92, Lemma 93, Lemma 94, Lemma 95 and Lemma 96 together with Lemma 84, Theorem 85 and Lemma 86 finish the proof of Theorem 64.

Explicit Construction. We give an explicit construction of an integrality gap instance for $\Gamma_5 = (P, Q), P = \mathrm{Ham}_k\{1, k\}, Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{b\}$ for arbitrary $b \in \{0, 1, \ldots, k\}$. Let L be a large constant (depends on k, to be set later). We have $n = 2k - 1 + \binom{2k-1}{k} L$ variables $x_i : i \in [2k-1], x_S^{(i)} : i \in [L], S \subseteq [2k-1], |S| = k$. Our constraints are the following: for every subset $S \subseteq [2k]$ with $|S| = k$ and $S = \{i_1, i_2, \ldots, i_k\}$, we pick L new variables

$x_S^{(1)}, x_S^{(2)}, \ldots, x_S^{(L)}$. The constraints are

$$\{x_{i_1}, x_{i_2}, \ldots, x_{i_k}\}, \{x_{i_2}, x_{i_3}, \ldots, x_{i_k}, x_S^{(1)}\}, \{x_{i_3}, \ldots, x_S^{(1)}, x_S^{(2)}\}, \ldots, \{x_S^{(L-k+1)}, \ldots, x_S^{(L)}\},$$
$$\{x_S^{(L-k+2)}, \ldots, x_S^{(L)}, \overline{x_{i_1}}\}, \{x_S^{(L-k+3)}, \ldots, \overline{x_{i_1}}, \overline{x_{i_2}}\}, \ldots, \{\overline{x_{i_1}}, \overline{x_{i_2}}, \ldots, \overline{x_{i_k}}\}.$$

We choose L to be the constant factor from Lemma 99 with $\alpha = \frac{k-3}{k-1}$. The idea is that when all the variables in the constraint $\{x_{i_1}, x_{i_2}, \ldots, x_{i_k}\}$ are all set to be True, all the variables in $\{\overline{x_{i_1}}, \overline{x_{i_2}}, \ldots, \overline{x_{i_k}}\}$, and as there are a series of constraints between them where we alter a single variable, there must exist a constraint where there are exactly b variables that are set to True.

Formally, we show that this instance does not satisfy $Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{b\}$. Suppose for contradiction that there is an assignment that satisfies all the constraints. Since there are $2k-1$ variables $x_1, x_2, \ldots, x_{2k-1}$, at least k of them are set to be true. If not, then at least k of the negated variables are set to be true. This implies that there is a sequence of constraints where the endpoints are assigned all True and all False, and at every point, we change a single variable. This implies that there is a constraint where there are exactly b variables that are set to True, a contradiction.

We now show that the instance has a basic SDP solution with zero error. Set $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{2k-1} \in \mathbb{S}^n$ to the variables $x_1, x_2, \ldots, x_{2k-1}$ such that $\mathbf{v}_i \cdot \mathbf{v}_0 = 0$, and $\mathbf{v}_i \cdot \mathbf{v}_j = \alpha$ for every $i \neq j$. Such a set of vectors is guaranteed to exist as n is large enough, and $\alpha \geq 0$. Finally, we use Lemma 99 to set the vectors $\mathbf{v}_S^{(i)}$ for every $S \subseteq [2k-1], |S| = k, i \in [L]$.

6.6 The SDP minion

As previously mentioned, polymorphisms are a powerful tool for understanding the computational complexity of PCSPs. However, beyond some of the simplest classes of PCSPs, individually classifying the complexity based on specific polymorphisms can be unwieldy. Instead, one often looks to higher-level structure between classes of polymorphisms, which is captured by the notion of *minions* (also called clonoids) [Bar+21].

Definition 100. *A minion $\mathcal{M}$ is a family of sets $\mathcal{M}_1, \mathcal{M}_2, \ldots$, where $\mathcal{M}_i$ are the objects are arity i. For every pair of positive integers i and j and map $\pi : [i] \to [j]$, there exists a map $^{/\pi} : \mathcal{M}_i \to \mathcal{M}_j$ known as a* minor map. *Further, these minor maps commute: $(f^{\pi})^{/\pi'} = f^{/\pi' \circ \pi}$.*

The most commonly discussed minion is the polymorphisms $\mathrm{Pol}(\Gamma)$ of a PCSP Γ. In this case the minors correspond to the identification of coordinates. Given a function f of arity i and a map $\pi : [i] \to [j]$, we have that

$$f^{/\pi}(x_1, \ldots, x_j) = f(x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(i)}).$$

It is straightforward to verify that the minor maps commute in the specified manner.

However, the 'objects' of a minion need not correspond to mathematical functions. The following are a few known examples:

- The *trivial minion* $\mathcal{M}_{triv}$ has that every arity has a single object: $\mathcal{M}_i = \{e\}$. All minor maps are thus the "trivial" map.

- The *dictator minion* (or projection minion) $\mathcal{M}_{dict}$ has each $\mathcal{M}_i = [i]$. The minor maps are then application of π: $k^{/\pi} = \pi(k)$.
- The *Basic LP minion* $\mathcal{M}_{BLP}$ has each $\mathcal{M}_i = \{(p_1, \ldots, p_i) : p_1, p_2, \ldots, p_i \geq 0, p_1 + \cdots + p_i = 1\}$ be the probability distributions on i elements. The minor maps combine elements of the probability distribution which map to the same value. That is,

$$(p_1, \ldots, p_i)^{/\pi} = \left(\sum_{\pi(k)=1} p_k, \ldots, \sum_{\pi(k)=j} p_k \right).$$

In order to better understand the complexity of PCSPs, we relate the polymorphic minions to other minions like the ones mentioned above. We can determine the relationship between minions via *minion homomorphisms*.

Definition 101. *A* minion homomorphism $\psi : \mathcal{M} \to \mathcal{N}$ *between two minions consists of maps* $\psi_i : \mathcal{M}_i \to \mathcal{N}_i$ *such that these maps commute with the respective minor maps of* $\mathcal{M}$ *and* $\mathcal{N}$*. That is, for all* $f \in \mathcal{M}_i$ *and* $\pi : [i] \to [j]$*, we have that*

$$\psi_j(f^{/\pi}) = \psi_i(f)^{/\pi}$$

The key theorem about the Basic LP minion is the following.

Theorem 102 ([Bar+21])**.** *The Basic LP solves a PCSP* Γ *if and only if* $\mathcal{M}_{BLP} \mapsto \mathrm{Pol}(\Gamma)$.

6.6.1 SDP Minion Definition

Like how the the BLP minion corresponds to probability distributions, the SDP minion we construct corresponds to SDP vectors. Similar techniques have been used in other minion constructions [CŽ22a].

Let $\mathbb{R}^\omega$ be infinite sequences of real numbers which are eventually 0 (and thus can be thought of as the union $\mathbb{R}^1 \cup \mathbb{R}^2 \cup \mathbb{R}^3 \cup \ldots$). It's not hard to see that $\mathbb{R}^\omega$ is an inner product space.

We now define the minion $\mathcal{M}_{\mathrm{SDP}}$ whose k-arity symbol is a list of k vectors $(\mathbf{w}_1, \ldots, \mathbf{w}_k)$ in $\mathbb{R}^\omega$. When convenient, we shall think of the whole object as a matrix $W \in \mathbb{R}^{k \times \omega}$.

We impose the following conditions on the vectors:

1. For all $i, j \in [k]$ with $i \neq j$, $\mathbf{w}_i \cdot \mathbf{w}_j = 0$.
2. $\sum_{i \in [k]} \|\mathbf{w}_i\|_2^2 = 1$.

Observe that the second condition is equivalent to $\|\sum_{i \in [k]} \mathbf{w}_i\|_2 = 1$.

The minors of $\mathcal{M}_{\mathrm{SDP}}$ are not too surprising, given $W \in \mathcal{M}_{\mathrm{SDP}}^{(k)}$ and a map $\pi : [k] \to [\ell]$, $W^{/\pi}$ is the matrix in $W' \in \mathbb{R}^{\ell \times \omega}$ where $\mathbf{w}'_i := \sum_{j \in \pi^{-1}(i)} \mathbf{w}_j$.

Claim 103. $\mathcal{M}_{\mathrm{SDP}}$ *is a minion.*

Proof. First, for each $W \in \mathcal{M}_{\mathrm{SDP}}^{(k)}$ and a map $\pi : [k] \to [\ell]$ we verify that $W' := W^{/\pi} \in \mathcal{M}_{\mathrm{SDP}}^{(\ell)}$.

First, fix $i \neq i' \in [\ell]$. We have that

$$\begin{aligned}\mathbf{w}'_i \cdot \mathbf{w}'_{i'} &= \left(\sum_{j \in \pi^{-1}(i)} \mathbf{w}_j\right) \cdot \left(\sum_{j' \in \pi^{-1}(i')} \mathbf{w}_{j'}\right)\\ &= \sum_{\substack{j \in \pi^{-1}(i)\\ j' \in \pi^{-1}(i')}} \mathbf{w}_j \cdot \mathbf{w}_{j'}\\ &= 0\end{aligned}$$

Further,

$$\left(\sum_{i' \in [\ell]} \mathbf{w}'_{i'}\right) \cdot \left(\sum_{i' \in [\ell]} \mathbf{w}'_{i'}\right) = \left(\sum_{i \in [k]} \mathbf{w}_i\right) \cdot \left(\sum_{i \in [k]} \mathbf{w}_i\right) = 1.$$

Thus, $W^{/\pi} \in \mathcal{M}^{(\ell)}_{\mathrm{SDP}}$.

The only remaining condition to check is that minors commute. Consider $\pi : [a] \to [b]$ and $\eta : [b] \to [c]$. Also consider. Let $U \in \mathcal{M}^{(a)}_{\mathrm{SDP}}$, $V \in \mathcal{M}^{(b)}_{\mathrm{SDP}}$, and $W \in \mathcal{M}^{(c)}_{\mathrm{SDP}}$ such that $V = U^{/\pi}$ and $W = V^{/\eta}$. We seek to verify that $W = U^{/(\eta\circ\pi)}$. For all $i \in [c]$, we have that

$$\begin{aligned}\mathbf{w}_i &= \sum_{i' \in \eta^{-1}(i)} \mathbf{v}_{i'}\\ &= \sum_{i' \in \eta^{-1}(i)} \sum_{i'' \in \pi^{-1}(i')} \mathbf{u}_{i''}\\ &= \sum_{i'' \in (\eta\circ\pi)^{-1}(i)} \mathbf{u}_{i''},\end{aligned}$$

as desired. □

The goal of this section is to prove the following theorem

Theorem 104. *If the exact*[13] *basic SDP solves* $\mathrm{PCSP}(\mathbf{A}, \mathbf{B})$ *if and only if* $\mathcal{M}_{\mathrm{SDP}} \to \mathrm{Pol}(\mathbf{A}, \mathbf{B})$

6.6.2 An alternative Basic SDP

We now present a modified basic SDP for which it is easier to make our arguments. We shall use notation similar to that of the [CŽ22a].

Let $(\mathbf{A}, \mathbf{B})$ be a PCSP template. Let $\mathbf{X}$ be an instance of $\mathrm{PCSP}(\mathbf{A}, \mathbf{B})$. We let A be the domain of $\mathbf{A}$, etc., so X is the variables of $\mathbf{X}$. We let $R^{\mathbf{A}}$ be the set of relations of $\mathbf{A}$ (or "clause types"). We let $R^{\mathbf{X}}$ be the constraints of $\mathbf{X}$.

[13] By 'exact' we mean an SDP solution that perfectly satisfies all the constraints. This is not feasible in principle as infinite precision is needed in the computed SDP matrix (c.f., [Fre04]). In practice, SDP solutions can be computed to $\mathrm{poly}(n)$ bits of precision precision, which we would consider to be a $1 - 1/2^{\mathrm{poly}(n)}$-robust solution. For PCSPs, it is unknown if there is a difference between exact, $1 - 1/2^{\mathrm{poly}(n)}$-robust and $1 - O(1)$ robust.

A basic SDP solution to $\mathbf{X}$ for $\mathrm{PCSP}(\mathbf{A}, \mathbf{B})$ is a collection of vectors $\mathbf{u}_{x,a} \in \mathbb{R}^\omega$ for all $x \in X, a \in A$ as well as $\mathbf{v}_{\mathbf{x},\mathbf{a}} \in \mathbb{R}^\omega$ for $\mathbf{x} \in R^{\mathbf{X}}$ and $\mathbf{a} \in s^R(\mathbf{x})$ (which is defined to be the set of the valid assignments to the clause $\mathbf{x}$) with the following properties:[14]

1. For all $x \in X$, $U_{\mathbf{x}} := (\mathbf{u}_{x,a} : a \in A) \in \mathcal{M}_{\text{SDP}}^{(A)}$.
2. For all $\mathbf{x} \in R^{\mathbf{X}}$, $V_{\mathbf{x}} := (\mathbf{v}_{\mathbf{x},\mathbf{a}} : \mathbf{a} \in s^R(\mathbf{x})) \in \mathcal{M}_{\text{SDP}}^{(A)}$.
3. For all $\mathbf{x} \in \mathbb{R}^{\mathbf{X}}$ with arity k and $i \in [k]$ and $a \in A$, we have that

$$\mathbf{u}_{\mathbf{x}_i,a} = \sum_{\substack{\mathbf{a} \in s^R(\mathbf{x}) \\ \mathbf{a}_i = a}} \mathbf{v}_{\mathbf{x},\mathbf{a}}.$$

Equivalence with traditional basic SDP

The traditional basic SDP does not specify vectors $\mathbf{v}_{\mathbf{x},\mathbf{a}}$. Rather, the traditional basic SDP keeps track of a probability distribution $\lambda_{\mathbf{x}}$ on assignments to $\mathbf{x}$. And for any pairs of variable, assignment pairs $(x_i, a_i), (x_j, a_j)$, we have that

$$\mathbf{u}_{x_i,a_i} \cdot \mathbf{u}_{x_j,a_j} = \Pr_{\mathbf{a} \sim \lambda_{\mathbf{x}}}[\mathbf{a}_i = a_i \wedge \mathbf{a}_j = a_j].$$

Clearly any solution to our Basic SDP is a solution to that basic SDP, as the probability $\lambda_{\mathbf{x}}(\mathbf{a})$ is precisely $\|\mathbf{v}_{\mathbf{x},\mathbf{a}}\|^2$, and the above linear condition checks out as a property of the vectors.

The other direction is more tricky. Assume we have a traditional basic SDP solution. Recall that each $\mathbf{u}_{x,a} \in \mathbb{Q}^\omega$. Since there are finitely many vectors, there exists some $N \in \mathbb{N}$ such that each $\mathbf{u}$ has its support within the first N coordinates. Provisionally assign $\hat{\mathbf{v}}_{\mathbf{x},\mathbf{a}}$ to be $\sqrt{\lambda_{\mathbf{x}}(\mathbf{a})} \cdot e_i$, where $i > N$ are chosen uniquely for each clause-assignment pair. As there are finitely many clause-assignment pairs, all of these are in $\mathbb{Q}^\omega$.

However, it is obvious the $\hat{\mathbf{v}}$'s won't be compatible with the $\mathbf{u}$'s as they are in trivially-intersecting vector spaces. However, we can define over all suitable choices of $i \in [k]$ and $a \in A$,

$$\hat{\mathbf{u}}_{\mathbf{x}_i,a} := \sum_{\substack{\mathbf{a} \in s^R(\mathbf{x}) \\ \mathbf{a}_i = a}} \hat{\mathbf{v}}_{\mathbf{x},\mathbf{a}}$$

It is not hard to see that $\mathbf{u}_{x_i,a_i} \cdot \mathbf{u}_{x_j,a_j} = \hat{\mathbf{u}}_{x_i,a_i} \cdot \hat{\mathbf{u}}_{x_j,a_j}$ for all suitable choices of x_i, a_i, x_j, a_j.

For a fixed $\mathbf{x}$, let $\hat{V}_{\mathbf{x}} \subseteq \mathbb{Q}^\omega$ be the subspace spanned by $\hat{\mathbf{v}}_{\mathbf{x},\mathbf{a}}$ and $\mathbf{u}_{x,a}$ for the $x \in \mathbf{x}$. Note that each $\hat{\mathbf{u}}$ is also in $\hat{V}_{\mathbf{x}}$. Since the $\hat{\mathbf{u}}$'s and the $\mathbf{u}$'s have the same dot products, there is a rigid rotation $\psi : \hat{V}_{\mathbf{x}} \to \hat{V}_{\mathbf{x}}$ which sends each $\hat{\mathbf{u}}_{(x,a)}$ to $\mathbf{u}_{(x,a)}$.

Define $\mathbf{v}_{\mathbf{x},\mathbf{a}} = \psi(\hat{\mathbf{v}}_{\mathbf{x},\mathbf{a}})$. Then, observe that

$$\mathbf{u}_{\mathbf{x}_i,a} = \psi(\hat{\mathbf{u}}_{\mathbf{x}_i,a}) = \sum_{\substack{\mathbf{a} \in s^R(\mathbf{x}) \\ \mathbf{a}_i = a}} \psi(\hat{\mathbf{v}}_{\mathbf{x},\mathbf{a}}) = \sum_{\substack{\mathbf{a} \in s^R(\mathbf{x}) \\ \mathbf{a}_i = a}} \mathbf{v}_{\mathbf{x},\mathbf{a}},$$

as desired. Thus, the two SDP formulations are equivalent.

[14] Usually, there is a special vector $\mathbf{v}_0$, but we can omit it without changing the power of the algorithm. This will be more convenient for the analysis.

6.6.3 From minion homomorphism to SDP rounding algorithm

First we show the "easy" direction that the minion homomorphism implies that the basic SDP solves $\mathrm{PCSP}(\mathbf{A}, \mathbf{B})$.

Theorem 105. *If* $\mathcal{M}_{\mathrm{SDP}} \to \mathrm{Pol}(\mathbf{A}, \mathbf{B})$*, then the basic SDP solves* $\mathrm{PCSP}(\mathbf{A}, \mathbf{B})$.

Proof. Fix $\mathbf{X}$ and an exact SDP solution $M^{\mathbf{X}}$ with corresponding vectors $\mathbf{u}_{x,a}$ and $\mathbf{v}_{\mathbf{x},\mathbf{a}}$ (and $U_x, V_{\mathbf{x}}$) with the prescribed properties. Let $\psi : \mathcal{M}_{\mathrm{SDP}} \to \mathrm{Pol}(\mathbf{A}, \mathbf{B})$ be the minion homomorphism.

Define the assignment $f : X \to B$ to be $\psi(U_x)(a : a \in A)$. Unpacking this, $\psi(U_x) \in \mathrm{Pol}(A, B)$ is of arity A, so we just plug in the coordinates of A listed in a canonical order. For any clause $\mathbf{x}$, we know that $\mathbf{b} := \psi(V_{\mathbf{x}})(\mathbf{a} : \mathbf{a} \in s^R(\mathbf{x}))$ satisfies $R^{\mathbf{B}}$. Thus, it suffices to show that $\mathbf{b}_i = f(\mathbf{x}_i)$ for all $i \in k$.

Let $\pi_i : R^{\mathbf{A}} \to A$ which maps $\mathbf{a}$ to $\mathbf{a}_i$. It is straightforward to show that condition (3) of the basic SDP implies that $U_x = (V_{\mathbf{x}})_{/\pi_i}$. Thus, since ψ is a minion homomorphism,

$$f(\mathbf{x}_i) = \psi(U_x)(a : a \in A) = \psi(V_{\mathbf{x}})_{/\pi_i}(a : a \in A) = \psi(V_{\mathbf{x}})_i(\mathbf{a} : \mathbf{a} \in s^R(\mathbf{x})) = \mathbf{b}_i,$$

as desired. Thus, the exact basic SDP solves $\mathrm{PCSP}(\mathbf{A}, \mathbf{B})$. □

6.6.4 From SDP rounding algorithm to minion homomorphism

We now prove the converse.

Theorem 106. *If the basic SDP solves* $\mathrm{PCSP}(\mathbf{A}, \mathbf{B})$*, then* $\mathcal{M}_{\mathrm{SDP}} \to \mathrm{Pol}(\mathbf{A}, \mathbf{B})$

Proof. We adopt the proof technique of [CŽ22a]. Let $\mathcal{F} \subset \mathcal{M}_{\mathrm{SDP}}$ be any finite subset. Let $\mathbf{F}$ be an instance of $\mathrm{PCSP}(\mathbf{A}, \mathbf{B})$ whose variables $F = \mathcal{F} \cap \mathcal{M}_{\mathrm{SDP}}^{(A)}$ – the arity-A elements of $\mathcal{F}$. The clauses $R^{\mathbf{F}}$ are on k-tuples $(W_1, \ldots, W_k)$ of F for which there is $Z \in \mathcal{F} \cap \mathcal{M}_{\mathrm{SDP}}^{(R^{\mathbf{A}})}$ with the following property. For all $i \in [k]$, let $\pi_i : R^{\mathbf{A}} \to A$ be the ith coordinate projection map. Then, $W_i = Z_{/\pi}$.

If we can show that the rational basic SDP solves F, then we know that $\mathbf{F} \to \mathbf{B}$ (by definition of the rational basic SDP solving $\mathrm{PCSP}(\mathbf{A}, \mathbf{B})$. Then, via a argument (e.g., like the De Brujin-Erdos theorem [BE51], for more details see Remark 7.13 of [Bar+21] or [CŽ22a], etc.) this implies that *free structure*[15] $\mathbb{F}_{\mathcal{M}_{\mathrm{SDP}}}(\mathbf{A}) \to \mathbf{B}$ which implies that $\mathcal{M}_{\mathrm{SDP}} \to \mathrm{Pol}(\mathbf{A}, \mathbf{B})$. Thus, it suffices to construction a rational basic SDP solution to $\mathbf{F}$.

The remainder of the proof writes itself. For every $W \in F$ and $a \in A$, let $\mathbf{u}_{W,a} = \mathbf{w}_a$ (the ath column of W). Likewise, for every clause τ on $(W_1, \ldots, W_k)$ via $Z \in \mathcal{F} \cap \mathcal{M}_{\mathrm{SDP}}^{(R^{\mathbf{A}})}$ and $\mathbf{a} \in s^R(Z)$ (that is a valid solution to the clause indexed by Z), we let $\mathbf{v}_{\tau,\mathbf{a}} = Z_{\mathbf{a}}$.

We then need to check conditions 1-3. Conditions 1 and 2 are verbatim from $W, Z \in \mathcal{M}_{\mathrm{SDP}}$. Condition 3 is precisely that $W_i = Z_{/\pi_i}$.

Since all the vectors have rational coordinates, the dot product matrix of these vectors is a rational basic SDP solution. That is, the rational basic SDP solves $\mathbf{F}$, so $\mathcal{M}_{\mathrm{SDP}} \to \mathrm{Pol}(\mathbf{A}, \mathbf{B})$. □

[15] See [Bar+21] for a precise definition.

This completes the proof of Theorem 104

6.7 Missing Proofs

Proof of Lemma 79. We first consider the case when $k = 1$. Without loss of generality, let $P = Q = \{+1\}$, and we use $\mathbf{v}_1 = \alpha\mathbf{v}_0 + \mathbf{v}'_1$ to denote the SDP vector corresponding to the variable used in the constraint. As the basic SDP has error at most $\sqrt{\epsilon}$, we get that

$$\alpha_1 \geq 1 - \sqrt{\epsilon}$$

As $\alpha_1^2 + \|v'_1\|_2^2 = 1$, $\|v'_1\| \leq O(\epsilon^{0.25})$. Thus, using Proposition 71, we get that $\langle \zeta, \mathbf{v}'_1 \rangle \leq O(\epsilon^{0.25} r)$ with probability at least $1 - e^{\frac{-r^2}{2}} \geq 1 - \sqrt{\epsilon}$. On the other hand, using Proposition 70, we get that $|\langle \zeta, \mathbf{v}_0 \rangle| \geq \frac{1}{r}$ with probability at least $1 - \frac{1}{r}$. This implies that

$$\delta\alpha_1 |\langle \zeta, \mathbf{v}_0 \rangle| \geq \frac{\delta}{2r}$$

Thus, with probablity at least $1 - O(\frac{1}{r})$, we have

$$\langle \zeta, \mathbf{v}'_1 \rangle \leq O(\epsilon^{0.25} r) < \frac{\delta}{2r} \leq \delta\alpha_1 |\langle \zeta, \mathbf{v}_0 \rangle|$$

Hence, with probability at least $1 - O(\frac{1}{r})$, we round the variable to $+1$.

We now consider the general case when $k \geq 2$. Note that the above proof for $k = 1$ holds when $P = \mathrm{Ham}_k\{0\}$ or when $P = \mathrm{Ham}_k\{k\}$. We are left with the setting when $P = \mathrm{Ham}_k\{0, k\}$. In order to show that our algorithm is a robust algorithm for this PCSP, it suffices to show that all the elements in the predicates are rounded to the same value with high probability. Consider $i, j \in [k]$. We show that the probability that the variables x_i and x_j get rounded to different values is at most $O\left(\frac{1}{r}\right)$. Using the union bound over all the $\binom{k}{2}$ pairs of indices, we get our required claim.

We first collect useful properties using the fact that the basic SDP is supported with probability at least $1 - c$ on P.

1. (First moment.) We have
$$|\mu_i - \mu_j| \leq 2c.$$
2. (Second moment.) We have
$$\langle \mathbf{v}_i, \mathbf{v}_j \rangle \geq 1 - 2c$$
   Using this, we get
$$\begin{aligned} \left\|\mathbf{v}'_i - \mathbf{v}'_j\right\|_2^2 &= \mathbf{v}_i - \mathbf{v}_j + (\alpha_j - \alpha_i)\mathbf{v}_0 \\ &\leq \|\mathbf{v}_i - \mathbf{v}_j\|_2^2 + (\alpha_i - \alpha_j)^2 \\ &\leq O(c). \end{aligned}$$
   Thus, $\left\|\mathbf{v}'_i - \mathbf{v}'_j\right\| \leq O(\sqrt{c})$.

As earlier, we assume that c is at most $\sqrt{\epsilon}$.

Recall that our goal is to upper bound the probability that x_i and x_j are rounded to different values. Without loss of generality, suppose that x_i is rounded to $+1$, and x_j is rounded to -1. We get that

$$\langle \zeta, \mathbf{v}'_i \rangle \geq \delta \alpha_i |\langle \zeta, \mathbf{v}_0 \rangle|$$
$$\langle \zeta, \mathbf{v}'_j \rangle < \delta \alpha_j |\langle \zeta, \mathbf{v}_0 \rangle|$$

Using Proposition 71, we can infer that

$$|\langle \zeta, \mathbf{v}'_i \rangle - \langle \zeta, \mathbf{v}'_j \rangle| \leq O(r\sqrt{c})$$

with probability at least $1 - \frac{1}{r}$.

We consider two cases: first, when $|\alpha_i| \leq \frac{1}{2}$. As $\alpha_i^2 + \|\mathbf{v}'_i\|_2^2 = 1$, and $|\alpha_i - \alpha_j| \leq 2c$, we get that $\left\|\mathbf{v}'_j\right\| = \Omega(1)$. In this case, we have

$$\begin{aligned}\langle \zeta, \mathbf{v}'_j \rangle &\geq \langle \zeta, \mathbf{v}'_i \rangle - O(r\sqrt{c}) \\ &\geq \delta \alpha_i |\langle \zeta, \mathbf{v}_0 \rangle| - O(r\sqrt{c})\end{aligned}$$

Thus, we have

$$\langle \zeta, \mathbf{v}'_j \rangle \in [\delta \alpha_i |\langle \zeta, \mathbf{v}_0 \rangle| - O(r\sqrt{c}), \delta \alpha_j |\langle \zeta, \mathbf{v}_0 \rangle|]$$

Here, $\langle \zeta, \mathbf{v}'_j \rangle \in [p, q]$ where $q - p \leq O(\delta r) + O(r\sqrt{c}) \leq O(\delta r)$. However, as $\left\|\mathbf{v}'_j\right\| \geq \Omega(1)$, this happens with probability at most $O(\delta r)$.

Now, suppose that $|\alpha_i| \geq \frac{1}{2}$. We have

$$\delta \alpha_i |\langle \zeta, \mathbf{v}_0 \rangle| \geq \delta \alpha_j |\langle \zeta, \mathbf{v}_0 \rangle| - 2\delta c |\langle \zeta, \mathbf{v}_0 \rangle|$$

However, as $\langle \zeta, \mathbf{v}_0 \rangle \sim \mathcal{N}(0, 1)$, we have $|\langle \zeta, \mathbf{v}_0 \rangle| \leq r$ with probability at least $1 - \sqrt{\epsilon}$. Thus, with probability at least $1 - \sqrt{\epsilon}$, we have

$$\langle \zeta, \mathbf{v}'_i \rangle \geq \delta \alpha_i |\langle \zeta, \mathbf{v}_0 \rangle| \geq \langle \zeta, \mathbf{v}'_j \rangle - 2\delta c r$$

We have $\delta \alpha_i |\langle \zeta, \mathbf{v}_0 \rangle| \in [p, q]$ where $q - p \leq O(\delta c r) + O(r\sqrt{c})$. However, this happens with probability at most $O(\frac{r\sqrt{c}}{\delta}) \leq O(\frac{1}{r})$. □

Proof of Lemma 82. Suppose that $\mathbf{a} = \mathrm{sgn}(\mathbf{x} - \mathbf{y})$ for $\mathbf{x}, \mathbf{y} \in \mathrm{Aff}(P)$, and $x_i \neq y_i \forall i \in [k]$. By modifying the affine combinations slightly, we can assume that $\mathbf{x}$ and $\mathbf{y}$ are a rational affine combinations of P while still preserving the fact that $\mathbf{a} = \mathrm{sgn}(x - y)$. . In other words, there exist $p_1, p_2, \ldots, p_K, q_1, q_2, \ldots, q_K \in \mathbb{Q}$ such that $\sum_{i \in [K]} p_i = \sum_{i \in [K]} q_i = 1$, and $\mathbf{x} = \sum_{i \in [K]} p_i \mathbf{a}_i$, $\mathbf{y} = \sum_{i \in [K]} q_i \mathbf{a}_i$, where $\{-1, +1\}^k = \{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_K\}$. Let N be a positive integer such that we can write $p_i = \frac{p'_i}{N}, q_i = \frac{q'_i}{N}$ where p'_i, q'_i are integers for every $i \in [K]$.

Let S be a multiset of $\{-1, +1\}^k$ where we take union over all $i \in [K]$, p'_i copies of $\mathbf{a}_i$, assign them a sign $\mathrm{sgn}(p'_i)$, and q'_i copies of $\mathbf{a}_i$, assign them a sign $\mathrm{sgn}(-q'_i)$. As we have

$\sum_{i\in[K]} p'_i = \sum_{i\in[K]} q'_i$, we get that there are equal number of vectors in S that are assigned $+1$ sign and equal number of them that are assigned -1. Let $\mathbf{z}$ denote the signed sum of all vectors (including repetitions) in S. Note that $\mathrm{sgn}(\mathbf{z}) = \mathrm{sgn}(\mathbf{x}-\mathbf{y})$. As each element of $\mathbf{z}$ is an integer, we get that the absolute value of each coordinate in $\mathbf{z}$ is at least 1. Furthermore, we can take multiple copies of S to ensure that the absolute value of each coordinate in $\mathbf{z}$ is at least 2. Now, we add an arbitrary element of P with sign $+1$ to S. Note that we still have that the signed sum of S i.e., the updated $\mathbf{z}$ satisfies $\mathrm{sgn}(\mathbf{z}) = \mathrm{sgn}(\mathbf{x}-\mathbf{y})$. Furthermore, $\mathbf{z} = \mathbf{x}_1 - \mathbf{x}_2 + \ldots + \mathbf{x}_L$ where each $\mathbf{x}_i \in P$. Thus, $\mathbf{a} = \mathrm{sgn}(\mathbf{x}-\mathbf{y}) = \mathrm{sgn}(w) = \mathrm{sgn}(\mathbf{x}_1 - \mathbf{x}_2 + \ldots + \mathbf{x}_L) \in O_{AT}(P)$. Thus,

$$\{\mathrm{sgn}(\mathbf{x}-\mathbf{y}) : \mathbf{x},\mathbf{y} \in \mathrm{Aff}(P), \forall i, x_i \neq y_i\} \subseteq O_{AT}(P)$$

To prove the other direction, suppose that $\mathbf{a} \in O_{AT}(P)$. That is, $\mathbf{a} = \mathrm{sgn}(\mathbf{x}_1 - \mathbf{x}_2 + \ldots + \mathbf{x}_L)$. Let S be a multiset of $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_L$ with the corresponding sign as in the summation. As P is non-trivial in every coordinate i.e., for every $i \in [k]$, there exist assignments $\mathbf{x}$ in P where $x_i = +1$, and similarly, $\mathbf{x}' \in P$ where $x'_i = -1$. By adding vectors with both signs $+1$ and -1, we can assume that S is non-trivial in every coordinate while still preserving the fact that the sign vector of the signed sum of S is equal to $\mathrm{sgn}(\mathbf{a})$. We modify S while still preserving this property to ensure that the signed sum of vectors in S has absolute value at least 2 in every coordinate.

As there are odd number of vectors in S, the signed sum of the vectors has absolute value at least 1 in every coordinate. Fix a vector $\mathbf{x}_i \in S$. Create two copies of every other vector in S (with the same sign as the original). Note that this operation does not alter the sign vector of the signed sum of the vectors in S. We can repeat this process at most $2k$ times to ensure that in the final multiset S, the signed sum has absolute value at least 2 in every coordinate. Finally, we add an arbitrary vector with sign -1 to S, to ensure that there are equal number of vectors with $+1$ and -1 sign in S. Overall, we get that there are $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N \in P$ and $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N \in P$ such that

$$\mathbf{a} = \mathrm{sgn}(\mathbf{x}_1 + \ldots + \mathbf{x}_N - \mathbf{y}_1 - \ldots - \mathbf{y}_N) = \mathrm{sgn}\left(\frac{1}{N}\mathbf{x}_1 + \ldots + \frac{1}{N}\mathbf{x}_N - \frac{1}{N}\mathbf{y}_1 - \ldots - \frac{1}{N}\mathbf{y}_N\right)$$

Thus, we get that $\mathbf{a} \subseteq \{\mathrm{sgn}(\mathbf{x}-\mathbf{y}) : \mathbf{x},\mathbf{y} \in \mathrm{Aff}(P), \forall i, x_i \neq y_i\}$, completing the proof that

$$O_{AT}(P) \subseteq \{\mathrm{sgn}(\mathbf{x}-\mathbf{y}) : \mathbf{x},\mathbf{y} \in \mathrm{Aff}(P), \forall i, x_i \neq y_i\} \qquad \square$$

Proof of Lemma 84. We extensively use the properties of $\mathrm{AT}, \mathrm{MAJ}$ polymorphisms of symmetric PCSPs, and ppp-reductions between symmetric folded PCSPs proved in [BG21b].

Consider $(P,Q) \in \Gamma$ be of arity k such that $\mathrm{MAJ}_{L_1}, \mathrm{AT}_{L_2} \notin \mathrm{Pol}(\Gamma)$ for some odd integers L_1, L_2. Note that $P \nsubseteq \{(-1,-1,\ldots,-1),(+1,+1,\ldots,+1)\}$ as in that case $O_{\mathrm{MAJ}}(P) = P \subseteq Q$, contradicting the fact that $\mathrm{MAJ} \notin \mathrm{Pol}(P,Q)$. Thus, there exists $l \in \{1,2,\ldots,k-1\}$ such that $\mathrm{Ham}_k\{l\} \subseteq P$.

Case 1. We first consider the case when $P = \mathrm{Ham}_k\{l\}$. As $\mathrm{MAJ} \nsubseteq \mathrm{Pol}(P,Q)$, there exists $b \in \{0,1,\ldots,k\}$ such that $\mathrm{Ham}_k\{b\} \notin Q$ and $\mathrm{Ham}_k\{b\} \subseteq O_{\mathrm{MAJ}}(P)$.

Suppose that $b \notin \{0,k\}$. Let $Q' = \{-1,+1\}^k \setminus \mathrm{Ham}_k\{b\}$. By definition, $\mathrm{MAJ} \nsubseteq \mathrm{Pol}(P,Q')$. As $O_{\mathrm{AT}}(\mathrm{Ham}_k\{l\}) = \mathrm{Ham}_k\{1,2,\ldots,k-1\}$, we get that $\mathrm{AT} \nsubseteq \mathrm{Pol}(P,Q')$. Thus, we get (P,Q)

where $P = \mathrm{Ham}_k\{l\}$, $Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{b\}$ where $b \in \{1, 2, \ldots, k-1\} \setminus \{l\}$. Note that $\mathrm{MAJ}, \mathrm{AT} \notin \mathrm{Pol}(P, Q)$. We now relax this PCSP furthermore, updating P, Q, l, k, b while preserving the following two properties:

1. At every step, $P = \mathrm{Ham}_k\{l\}, Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{b\}$ where $b \in \{1, 2, \ldots, k-1\} \setminus \{l\}$.
2. $\mathrm{MAJ}, \mathrm{AT} \notin \mathrm{Pol}(P, Q)$.

As $O_{\mathrm{MAJ}}(P) = \mathrm{Ham}_k\{0, 1, \ldots, k\} \cap \{2l-k+1, \cdots, 2l-1\}$, and $b \in O_{\mathrm{MAJ}}(P)$, we get that $b \in \{2l-k+1, \cdots, 2l-1\} \cap \{0, \cdots, k\}$. Furthermore, as $b > 0$, we get that $l > 1$. Similarly, we get that $l < k-1$. This also implies that $k \geq 4$ as $l \in \{1, \ldots, k-1\}$.

We use the following two tools to relax the PCSP:

1. Given a PCSP $P = \mathrm{Ham}_k\{l\}, Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{b\}$ where $b \in \{1, 2, \ldots, k-1\} \setminus \{l\}$, then the PCSP $P' = \mathrm{Ham}_{k-1}\{l\}, Q = \mathrm{Ham}_{k-1}\{0, 1, \ldots, k\} \setminus \{b\}$ is a relaxation of (P, Q) (Claim 4.2 of [BG21b]). As long as $b < k-1$ and $b \neq 2l-k+1$, this relaxation preserves the above two properties.
2. Given a PCSP $P = \mathrm{Ham}_k\{l\}, Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{b\}$ where $b \in \{1, 2, \ldots, k-1\} \setminus \{l\}$, then the PCSP $P' = \mathrm{Ham}_{k-1}\{l-1\}, Q = \mathrm{Ham}_{k-1}\{0, 1, \ldots, k\} \setminus \{b-1\}$ is a relaxation of (P, Q) (Claim 4.4 of [BG21b]). As long as $b > 1$ and $b \neq 2l-1$, this relaxation preserves the above two properties.

Now, we relax the PCSP using the above two steps. As k is decreasing at every step, this procedure terminates at some point. Then, either of the two conditions hold:

1. $b = 1, b = 2l-k+1$. In this case, we get that $l = \frac{k}{2}$ and $b = 1$. Thus, (P', Q') is a relaxation of Γ where $P' = \mathrm{Ham}_k\{\frac{k}{2}\}, Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{1\}$ where k is even and is at least 4.
2. $b = k-1, b = 2l-1$. In this case, we get that $l = \frac{k}{2}$ and $b = k-1$. Thus, (P', Q') is a relaxation of Γ where $P' = \mathrm{Ham}_k\{\frac{k}{2}\}, Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{k-1\}$ where k is even and is at least 4.

Suppose that there is no $b \notin \{0, k\}$ such that $\mathrm{Ham}_k\{b\} \subseteq O_{\mathrm{MAJ}}(P) \setminus Q$. As $O_{\mathrm{MAJ}}(P) \nsubseteq Q$, by negating the variables if needed, we can assume that $\mathrm{Ham}_k\{0\} \in O_{\mathrm{MAJ}}(P) \setminus Q$. Furthermore, there exists $b \in \{1, 2, \ldots, k-1\}$ such that $\mathrm{Ham}_k \nsubseteq Q$ as $O_{\mathrm{AT}}(P) \nsubseteq Q$. Thus, we obtain a relaxation (P, Q) of the original PCSP such that $P = \mathrm{Ham}_k\{l\}, Q = \mathrm{Ham}_k\{1, \ldots, k\} \setminus \{b\}$ where $l, b \in \{1, 2, \ldots, k-1\}, b > 2l-1$. By using the first type of relaxation used above (Claim 4.2 of [BG21b]), we obtain a new relaxation such that $P = \mathrm{Ham}_k\{l\}, Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{0, k-1\}$ where $l \in \{1, 2, \ldots, k-1\}, l \leq \frac{k-1}{2}$.

Case 2. There exist $l \neq l'$ such that $\mathrm{Ham}_k\{l, l'\} \subseteq P$. Recall that $P \nsubseteq \mathrm{Ham}_k\{0, k\}$. This implies that $O_{\mathrm{AT}}(P) = \mathrm{Ham}_k\{0, 1, \ldots, k\}$. Hence, we can get a relaxation (P, Q') of the original PCSP such that $Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{b\}$ such that $\mathrm{Ham}_k\{b\} \notin O_{\mathrm{MAJ}}(P)$.

For ease of notation, let $P = \mathrm{Ham}_k S$, where $S \subseteq \{0, 1, \ldots, k\}$. First, consider the case when $\min S = 0, \max S = k$. As mentioned earlier, we know that there exists $l \in \{1, 2, \ldots, k-1\}$ such that $\mathrm{Ham}_k\{l\} \in P$. Thus, we can reduce the existing PCSP to (P, Q) where $P =$

$\mathrm{Ham}_k\{0, l, k\}, Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{b\}$ where $b \notin \{0, l, k\}$. We consider three cases separately:

1. Suppose that $l \leq \frac{k-1}{2}$. In this case, we have a relaxation (P, Q) where $P = \mathrm{Ham}_k\{l, k\}$ and $Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{b\}$ where $b \notin \{l, k\}$. Note that this relaxation does not contain AT or MAJ as polymorphisms.
2. Suppose that $l = \frac{k}{2}$. In this case, we have a relaxation (P, Q) where $P = \mathrm{Ham}_k\{l\}$ and $Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{b\}$ where $b \notin \{0, l, k\}$. This also doesn't have AT and MAJ as polymorphisms. Furthermore, we have shown earlier that we can relax this further to earlier mentioned three PCSPs.
3. Suppose that $l \geq \frac{k+1}{2}$. In this case, we have a relaxation (P, Q) where $P = \mathrm{Ham}_k\{0, l\}$ and $Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{b\}$ where $b \notin \{0, l\}$. Note that this relaxation does not contain AT or MAJ as polymorphisms.

Thus, we have a relaxation (P, Q) of the original PCSP where $P = \mathrm{Ham}_k\{l, l'\}, Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{b\}$ such that $l < l', \{l, l'\} \neq \{0, k\}, b \in O_{\mathrm{MAJ}}(P)$. We end up with the same relaxation when $\{\min S, \max S\} \neq \{0, k\}$.

If $\{l, l'\} = \{1, k\}$ or $\{0, k-1\}$, we get a relaxation of the original PCSP where $P = \mathrm{Ham}_k\{1, k\}, Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{b\}$, and we are done. If not, we get a series of relaxations of the original PCSP maintaining the two below properties:

1. $P = \mathrm{Ham}_k\{l, l'\}$ with $l < l'$ and $\{l, l'\} \neq \{0, k\}$ and $Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{b\}$. We also assume that $\{l, l'\} \neq \{1, k\}$ and $\{l, l'\} \neq \{0, k-1\}$.
2. $\mathrm{AT}, \mathrm{MAJ} \notin Pol(P, Q)$.

As with the earlier case, we update the PCSP using the two relaxations below:

1. We get $P' = \mathrm{Ham}_{k-1}\{l, l'\}$ and $Q = \mathrm{Ham}_{k-1}\{0, 1, \ldots, k\} \setminus \{b\}$ using Claim 4.2 of [BG21b]. For this to be a valid relaxation preserving the above properties, we need that $l' \neq k, b \neq k$ and $b \neq 2l - k + 1$.
2. We get $P' = \mathrm{Ham}_{k-1}\{l-1, l'-1\}$ and $Q = \mathrm{Ham}_{k-1}\{0, 1, \ldots, k\} \setminus \{b-1\}$ using Claim 4.4 of [BG21b]. For this to be a valid relaxation preserving the above properties, we need that $l \neq 0, b \neq 0$ and $b \neq 2l' - 1$.

As the arity of the predicates is decreasing at each step, this process terminates in finite steps. When we are unable to obtain a new relaxation using the above procedures, one of the following must be true.

1. $l' = k, b = 0$. In this case, we have a PCSP (P, Q) where $P = \mathrm{Ham}_k\{l, k\}$, $Q = \mathrm{Ham}_k\{1, 2, \ldots, k\}$, where $l \neq 0, l \leq \frac{k-1}{2}$.
2. $b = k, l = 0$. This can be relaxed to the above by negating the variables.
3. $b = k, b = 2l' - 1$. We have $l' = \frac{k+1}{2}$. In this case, we have PCSP (P, Q) where $P = \mathrm{Ham}_k\{l, \frac{k+1}{2}\}, Q = \mathrm{Ham}_k\{0, 1, 2, \ldots, k-1\}$, where $l \leq \frac{k-1}{2}$.
4. We have $b = 2l - k + 1, b = 0$. In this case, we can relax to the above by negating the variables.

Thus, we have relaxed the original PCSP into either of the following PCSPs.

1. k is even, and $P = \mathrm{Ham}_k\{\frac{k}{2}\}, Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{b\}$ where $b \in \{1, k-1\}$.
2. k is odd, $P = \mathrm{Ham}_k\{l, \frac{k+1}{2}\}$, $Q = \mathrm{Ham}_k\{0, 1, 2, \ldots, k-1\}$, where $l \leq \frac{k-1}{2}$.
3. $P = \mathrm{Ham}_k\{l, k\}$, $Q = \mathrm{Ham}_k\{1, 2, \ldots, k\}$, where $l \neq 0, l \leq \frac{k-1}{2}$.
4. $P = \mathrm{Ham}_k\{l\}, Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{0, k-1\}$ where $l \in \{1, 2, \ldots, k-1\}, l \leq \frac{k-1}{2}$.
5. $P = \mathrm{Ham}_k\{1, k\}, Q = \mathrm{Ham}_k\{0, 1, \ldots, k\} \setminus \{b\}$ for arbitrary b.

$\square$

Chapter 7

Revisiting Alphabet Reduction

7.1 Introduction

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

The PCP theorem was originally proved using algebraic techniques such as the low-degree test and the sum-check protocol. In a striking work, Dinur [Din07] gave an alternative combinatorial proof of the PCP theorem. Her proof works by amplifying the 'Unsat value' of a CSP instance — the fraction of constraints any assignment should violate. The goal is to show that it is NP-hard to distinguish if the Unsat value of a CSP instance is equal to 0 or at least a constant $c > 0$. Starting with a NP-hard problem such as 3-coloring with m constraints, we can already deduce that it is NP-hard to identify if Unsat value is equal to 0 or at least $1/m$. The Unsat value is increased slowly and iteratively via two steps — *gap amplification* and *alphabet reduction*. In gap amplification, we incur a constant factor blow up in the size of the instance, and get a constant factor improvement in the Unsat value. However, this step also blows up the alphabet size. To alleviate this, alphabet reduction brings back the alphabet size to an absolute constant while losing a constant factor in the Unsat value (and blows up the instance size by a constant factor). Combining both the steps, we can increase the Unsat value by a constant factor (say 2) incurring a constant factor blow up in the size of the instance. Repeating this $\log m$ times proves the PCP theorem.

In this chapter, we revisit the alphabet reduction step. Dinur implemented this step by an "inner" PCP construction, which is in effect a gadget reducing a specific predicate ψ to be tested to a collection Ψ of constraints over a fixed (say Boolean) alphabet, such that if ψ is unsatisfiable, then a constant fraction of constraints of Ψ must be violated by any assignment.[1] This inner PCP is then applied to all constraints in the CSP instance (say $\mathcal{G}$) produced by the gap amplification step. The collection of inner PCPs as such only ensure that each constraint of $\mathcal{G}$ is individually satisfiable, which is not very meaningful. To ensure that the inner PCPs together ensure that the constraints of $\mathcal{G}$ are all satisfiable by a *single consistent* assignment, error-correcting codes are used to encode the purported assignments to the variables of $\mathcal{G}$. The inner PCP is also replaced by an *Assignment Tester* that ascertains whether the specific assignment given by these encodings satisfies the predicate ψ being checked.

The key observation driving this work is that instead of designing the inner PCP for *arbitrary* constraints (as in Dinur's paper), we can first reduce the CSP instance $\mathcal{G}$ produced by gap amplification to a *Label Cover* instance. Label Cover is a special kind of CSP which has arity 2, and whose underlying relations are functions (so the value of one of the variables in each constraint is determined by the value taken by the other variable in that constraint). Conveniently for us, we also observe that Dinur's gap amplification step in fact already produces a CSP with this Label Cover structure, allowing us to skip the reduction step.[2] We can thus focus on alphabet reduction when the CSP we are reducing from has the Label Cover structure, and is over a fixed, albeit large, alphabet. We then follow the influential Label Cover and Long Code framework, originally proposed in [BGS98] and strengthened in [Hås01] and since then applied in numerous works on inapproximability, to reduce the CSP obtained from gap amplification, now viewed as Label Cover, to a Boolean CSP. Finally, we reduce the Boolean CSP back to a Label Cover instance (see Section 7.4) that can be plugged in as input to the gap amplification step.

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

Theorem 107. *There is a polynomial time reduction from Label Cover with soundness $1-\delta$ to a fixed template CSP with soundness $1-C\delta$ for an absolute constant $C>0$.*

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

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

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

- In [Hås01] and [BGS98], the objective is to prove optimal inapproximability, or at least to

[1]While this might seem circular, as this is what the PCP reduction is trying to accomplish in the first place, the key is that this inner reduction can be highly inefficient (even triply exponential blow up is okay!), as it is applied to a constraint of constant size.

[2]Technically, the gap amplification step produces a version of Label Cover whose constraints are *rectangular* rather than functions, but this is a minor difference that can be easily accommodated in reductions from Label Cover.

get good soundness. However, our present goal is to prove 'just' a nontrivial soundness. (On the other hand, we also start with high soundness Label Cover.) This allows us to use a very convenient test distribution leading to a simple analysis.

- We remark that a similar statement as Theorem 107 can be also deduced using [BGS98, Section 4.1.1]. We believe that the test presented in this chapter is more direct since we benefit from ideas in [Hås01].

- It is possible to perform alphabet reduction using the Hadamard code instead of the long code as described in [GO05; RS07]; the latter [RS07], similarly to our proof, avoids explicit use of Assignment Testers.

- Long code tests correspond exactly to testing whether a function is a *polymorphism* of the corresponding CSP, and as such corresponds to gadget reductions in the algebraic approach to CSP (see e.g. [BKW17]). The PCP theorem surpasses these algebraic (gadget) reductions; this is even more evident when extending the scope from CSPs to *promise constraint satisfaction problems* (PCSPs) as there are PCSPs that can be shown to be NP-hard by using PCP theorem via a natural reduction through Label Cover, but cannot be shown to be NP-hard using only algebraic reductions [AGH17; Bar+21]. In this sense, the present work shows that this strength of the PCP theorem comes from the Gap Amplification step.

Alphabet Reduction is an essential step in both the original proof of the PCP theorem as well as Dinur's proof and deserves further attention. Our proof of alphabet reduction bypasses the concept of Assignment Testers and is more intuitive in our opinion as it is *nothing but a gadget reduction*. Our proof is elementary using only Parseval's identity from Fourier Analysis over the hypercube. Dinur's analysis used the Friedgut-Naor-Kalai theorem [FKN02] about Boolean functions with most of the Fourier mass at level 1. We believe that this makes our proof more accessible to readers that are new to PCPs. We also hope that this material might be useful in teaching the proof of PCP theorem as it relies only on techniques that any such basic course would cover anyway.

Outline

We start by formally defining CSP, Label Cover, and other preliminaries in Section 7.2. Then, in Section 7.3, we prove the main reduction from Label Cover to CSP. In Section 7.4, we show how the reduction can be used in the alphabet reduction step of Dinur's proof.

7.2 Preliminaries

7.2.1 Rectangular relation and the long code

In this chapter, we view the Label Cover problem as a binary CSP with the relations being projections. More generally, we consider Label Cover instances where the relations are rectangular.

Definition 108 (Rectangular relation)**.** *A relation $R \subseteq A \times B$ is said to be rectangular if there is a set C and functions $\pi\colon A \to C$ and $\sigma\colon B \to C$ such that $(a,b) \in R$ if and only if $\pi(a) = \sigma(b)$. Equivalently, R is rectangular if for all $a, a' \in A$ and $b, b' \in B$ such that $(a,b) \in R$, $(a,b') \in R$, and $(a',b) \in R$, we have $(a',b') \in R$.*

Long code is often used in conjunction with the Label cover problem to obtain inapproximability results. Loosely speaking, the *long code* is the longest (error-correcting) code over the Boolean alphabet that does not repeat bits. It is constructed as follows: the long code is a Boolean code of length 2^n which encodes a value $i \in [n]$ into a tuple p_i whose k-th coordinate, $k < 2^n$, is the i-th least significant digit of k in binary.

The long code be also described in another way: we view a Boolean tuple of length 2^n as an n-ary function $p\colon \{0,1\}^n \to \{0,1\}$ (each coordinate of the tuple encodes one value of p). In this perspective, the code words of the long code are functions p_i defined as $p_i(x_1, \dots, x_n) = x_i$. These functions are often called *dictators*.

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

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

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

7.2.2 Boolean Fourier analysis

As usual in Boolean Fourier analysis, we treat TRUE as -1 and FALSE as $+1$. In particular, in this notation, 'negation' is expressed as $\neg x = -x$, 'xor' $x \oplus y$ is expressed as $x \oplus y = xy$, and 'or' is the expressed by the following function:

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

We will use all the same symbols to denote the coordinatewise (or bitwise) application of these functions to tuples, e.g. $(x_1, x_2) \oplus (y_1, y_2) = (x_1 y_1, x_2 y_2)$.

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

$$\chi_\alpha(x_1, \dots, x_n) = \prod_{i \in \alpha} x_i.$$

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

Definition 109 (Fourier expansion)**.** *Given a function $f\colon \{-1,+1\}^n \to \mathbb{R}$, we can thus write it uniquely as a linear combination of this basis—*

$$f = \sum_{\alpha \subseteq [n]} \hat{f}(\alpha) \chi_\alpha$$

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

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

Theorem 110 (Parseval's Identity)**.** *For each Boolean valued function f, i.e., $f\colon \{-1,+1\}^n \to \{-1,+1\}$,*

$$\sum_{\alpha \subseteq [n]} \hat{f}(\alpha)^2 = 1.$$

Connection to the long code

We remark, that the function $\chi_{\{i\}}$ corresponds to a valid long code: the function p_i encoding the value i. Also observe that there is a connection between the natural distance defined by the inner product $\langle f, g \rangle$ on Boolean functions and *relative Hamming distance* of f and g: This is thanks to the fact that if $x, y \in \{-1, 1\}$ then $x = y$ if and only if $xy = 1$, and consequently, the relative Hamming distance of f to the long code word $p_i = \chi_{\{i\}}$ can be expressed as $\big(1 - \hat{f}(i)\big)/2$. This means that the closest valid long code to a function f is the p_i for which $\hat{f}(i)$ is maximal.

These ideas are manifested in the common strategy in rounding a Boolean function f to a long code: First make sure that coefficients $\hat{f}(\alpha)$ for large sets α are small enough, then decode to a value i that belongs to a small-enough (ideally 1-element) set α with a large-enough $\hat{f}(\alpha)$.

7.3 Label Cover to CSP

This section describes our gadget reduction from Label Cover to $\mathrm{CSP}(R)$ where R is the 4-ary relation over Boolean domain defined as

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

Theorem 111. *There exists absolute constant C such that given a Label Cover instance (not necessarily bipartite) $G = (V, E, \Sigma, \Pi)$ with rectangular constraints, there is a reduction from G that outputs an instance I of $\mathrm{CSP}(R)$ such that $\mathrm{size}(I) = O(\mathrm{size}(G))$ and*

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

Since the domain of $\mathrm{CSP}(R)$ is Boolean, the above reduces from an alphabet Σ of arbitrary size to the alphabet of size 2. We note that the constant in $O(\mathrm{size}(G))$ above hides exponential dependency on $|\Sigma|$.

We describe the reduction as a probabilistic checker of a solution to G encoded using a long code, i.e., the proof contains for each $u \in V$ a word $f_u\colon \{-1, 1\}^{|\Sigma|} \to \{-1, 1\}$. In other words, we design the test in such a way that if $s\colon V \to \Sigma$ is a solution to G, then the assignment $f_u \mapsto p_{s(u)}$ passes the test. This will then immediately give the completeness of the reduction.

The test is as follows: Sample an edge $e = (u, v)$ from E uniformly at random, and then with equal probability do one of the following

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

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

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

To simplify some notation, we assume $\Sigma = [n]$. We also assume that the tables for f_u's are *folded* so f_u is forced to satisfy $f(-x) = -f(x)$. This is a standard technique. Such a folding is ensured by including only one variable of each pair $x, -x$, and if the test queries f_u at the bit corresponding to some x that is not included, the bit $f(-x)$ is queried instead, and the value is negated. As a consequence of this folding, we have to allow for negation of variables in $\mathrm{CSP}(R)$. An important and useful consequence of this is that all 'even' Fourier coefficients of f vanish, i.e., $\hat{f}(\beta) = 0$ for all β such that $|\beta|$ is even. We remark that folding can be avoided in the construction of the gadget. Nevertheless, it considerably simplifies the calculations below. Further, for calculations, it is useful to view R as a predicate $\rho\colon \{\pm 1\}^4 \to \{0,1\}$ defined as

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

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

Let us now describe the two probabilistic checkers.

7.3.1 Long code test

The long code test has on input a table of a function f ($= f_u$ or f_v), and it is supposed to check whether this function is a code word of the long code, i.e., there is i such that $f = p_i$. We design the test so that all these words pass with probability 1. Since we are only using the predicate R, this further limits possible checks. In fact, we include all checks of the form $f(x_1, x_2, x_3, x_4) \in R$ that are passed by all dictators.[3]

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

$$(f(x), f(x \oplus (\mu \vee y)), f(z), f(z \oplus (\mu \vee \neg y))) \in R. \tag{7.1}$$

[3] Any function that passes any such test with probability 1 is called a *polymorphism of* R, see also [BKW17].

Note that for all $x, y, z, \mu \in \{-1, 1\}$, $(x, x \oplus (\mu \vee y), z, z \oplus (\mu \vee \neg y)) \in R$. This implies that any dictator function passes the test with probability 1, and therefore provides the completeness of the test. We also note that this test can give some false positives, e.g. the function $-p_i \colon x \mapsto -p_i(x)$ passes the test with probability 1, but is not a long code word. It is in fact a negation of the word p_i. It can be checked that all functions that pass are either long code words, or their negations. In the decoding, we simply decode the above function f to i.

The following lemma bounds the probability that the test accepts in the means of the Fourier coefficients. We remark, that since we want to ensure that f is as close to a valid long code as possible, the probability should decrease as the coefficients $\hat{f}(\alpha)$ for $\alpha \neq \{i\}$ increase. Indeed, the lemma states that this is the case.

Lemma 112. *Assuming that f is folded, the probability the long code test accepts is at most*

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

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

$$\begin{aligned}
&\mathbb{E}_{x,y,z,\mu}\rho(f(x), f(x \oplus (\mu \vee y)), f(z), f(z \oplus (\mu \vee \neg y))) \qquad (7.2)\\
&= \mathbb{E}_{x,y,z,\mu}\left[1 - \frac{\big(f(x)f(x \oplus (\mu \vee y) + 1\big)\big(f(z)f(z \oplus (\mu \vee \neg y)) + 1\big)}{4}\right]\\
&= \frac{3}{4} - \frac{1}{4}\mathbb{E}_{x,y,\mu} f(x)f(x \oplus (\mu \vee y)) - \frac{1}{4}\mathbb{E}_{y,z,\mu} f(z)f(z \oplus (\mu \vee \neg y))\\
&\quad - \frac{1}{4}\mathbb{E}_{x,y,z,\mu} f(x)f(x \oplus (\mu \vee y))f(z)f(z \oplus (\mu \vee \neg y))
\end{aligned}$$

We further simplify this expression one term at a time.

$$\begin{aligned}
\mathbb{E}_{x,y,\mu} f(x)f(x \oplus (\mu \vee y)) &= \mathbb{E}_{x,y,\mu} \sum_{\alpha,\beta} \hat{f}(\alpha)\hat{f}(\beta)\chi_\alpha(x)\chi_\beta(x \oplus (\mu \vee y)) \qquad (7.3)\\
&= \sum_{\alpha,\beta} \hat{f}(\alpha)\hat{f}(\beta)\mathbb{E}_x[\chi_\alpha(x)\chi_\beta(x)]\mathbb{E}_{y,\mu}[\chi_\beta(\mu \vee y)]\\
&= \sum_\alpha \hat{f}(\alpha)^2 \mathbb{E}_{y,\mu}[\chi_\alpha(y \vee \mu)] = \sum_\alpha \hat{f}(\alpha)^2(-1/2)^{|\alpha|}.
\end{aligned}$$

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

$$\mathbb{E}_{y,z,\mu} f(z)f(z \oplus (\mu \vee \neg y)) = \sum_\alpha \hat{f}(\alpha)^2(-1/2)^{|\alpha|}. \qquad (7.4)$$

Moving to the next term, we get

$$\begin{aligned}
&\mathbb{E}_{x,y,z,\mu} f(x)f(x \oplus (\mu \vee y))f(z)f(z \oplus (\mu \vee \neg y))\\
&\quad = \sum_{\alpha,\beta} \hat{f}(\alpha)^2\hat{f}(\beta)^2\mathbb{E}_{y,\mu}\chi_\alpha(\mu \vee y)\chi_\beta(\mu \vee \neg y) = \sum_{\alpha\cap\beta=\emptyset} \hat{f}(\alpha)^2\hat{f}(\beta)^2(-1/2)^{|\alpha\cup\beta|}. \qquad (7.5)
\end{aligned}$$

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

$$\begin{aligned}\frac{3}{4} - \frac{1}{2}\sum_{\alpha} \hat{f}(\alpha)^2(-1/2)^{|\alpha|} - \frac{1}{4}\sum_{\alpha\cap\beta=\emptyset} \hat{f}(\alpha)^2\hat{f}(\beta)^2(-1/2)^{|\alpha\cup\beta|} \\ = 1 - \frac{1}{2}\sum_{\alpha} \hat{f}(\alpha)^2\big((-1/2)^{|\alpha|} + 1/2\big) - \frac{1}{4}\sum_{\alpha\cap\beta=\emptyset} \hat{f}(\alpha)^2\hat{f}(\beta)^2(-1/2)^{|\alpha\cup\beta|} \quad (7.6)\end{aligned}$$

where for the last equality, we used Parseval's identity. Further, we assumed that f is folded, and therefore $\hat{f}(\alpha) = 0$ for all α such that $|\alpha|$ is even. Restricting the sums to α and β with odd cardinality, and using that for such disjoint α and β, $|\alpha \cup \beta|$ is even, the last expression of (7.6) can be bounded from above by

$$1 - \frac{1}{2}\sum_{|\alpha|>1} \hat{f}(\alpha)^2\,(3/8) - \frac{1}{4}\sum_{\alpha\cap\beta=\emptyset} \hat{f}(\alpha)^2\hat{f}(\beta)^2(1/2)^{|\alpha\cup\beta|} \leq 1 - \frac{3}{16}\sum_{|\alpha|>1} \hat{f}(\alpha)^2 \quad (7.7)$$

which concludes the proof. □

7.3.2 Constraint test

The constraint test has on input tables for functions f and g corresponding to some u and v such that $(u, v) \in E$, and it is supposed to test (assuming f and g are correct long code words) whether the values these functions encode satisfy the constraint given by a rectangular relation Π_e. We construct the test in a similar way as the long code test: We test functions f and g in 4 bits in such a way that long code words encoding satisfying values pass. In contrast with the long code test, we do not include all such tests, but only a selection; in particular, we include only tests that query two values from each function.

We assume that the constraint relation Π_e is given by $\pi, \sigma\colon [n] \to [m]$ such that $(i, j) \in \Pi_e$ if and only if $\pi(i) = \sigma(j)$, and we denote by y^π the vector in $\{-1, 1\}^n$ such that $y^\pi(i) = y(\pi(i))$.

Constraint test. Given $f, g\colon \{-1, 1\}^n \to \{-1, 1\}$ to test against satisfying a constraint Π_e given by $(i, j) \in \Pi_e$ if and only if $\pi(i) = \sigma(j)$ for fixed $\pi, \sigma\colon [n] \to [m]$. Choose $x, z \in \{-1, 1\}^n$ and $y \in \{-1, 1\}^m$ uniformly at random, and test whether

$$(f(x), f(x \oplus y^\pi), g(z), g(z \oplus (\neg y)^\sigma)) \in R. \quad (7.8)$$

Note that if both f and g are dictators, say $f = p_i$ and $g = p_j$, such that $\pi(i) = \sigma(j) = k$ then the above test accepts with probability 1. Indeed, the tuple gets evaluated to

$$(x(i), (x \oplus y^\pi)(i), z(j), (z \oplus (\neg y)^\sigma)(j)) = (x(i), x(i) \oplus y(k), z(j), z(j) \oplus \neg y(k))$$

which is in R for all x, y and z. This provides the completeness of the test.

In the analysis below, we will use the following notation.

Definition 113. *Let $\alpha \subseteq [n]$ and $\pi\colon [n] \to [m]$, we denote by $\pi[\alpha]$ the subset of $[m]$ defined by $\pi[\alpha] = \{k : |\pi^{-1}(k) \cap \alpha|$ is odd$\}$.*

The goal of the constraint check is to ensure that functions f, g which are far from valid long codes that encode values satisfying the constraint pass with low probability. Unfortunately, the test gives a lot of false positives: it accepts any pair of functions χ_α and χ_β such that $\pi[\alpha] = \sigma[\beta]$ with probability 1.[4] This is nevertheless good-enough since the long code test provides that relevant α and β contain only one element, and $\pi[\{i\}] = \sigma[\{j\}]$ if and only if $\pi(i) = \sigma(j)$.

Naturally, the pairs of α and β with $\pi[\alpha] = \sigma[\beta]$ will appear in the analysis below. A useful fact that will simplify the computation below is that $\prod_{i\in\alpha} x_{\pi(i)} = \prod_{i\in\pi[\alpha]} x_i$, for all $x_1, \ldots, x_m \in \{-1, 1\}$, which implies that

$$\chi_\alpha(x^\pi) = \chi_{\pi[\alpha]}(x).$$

Lemma 114. *Given that both f and g are folded, the probability that the consistency test accepts is at most*

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

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

$$\frac{3}{4} - \frac{1}{4}\mathbb{E}_{x,y}[f(x)f(x \oplus y^\pi)] - \frac{1}{4}\mathbb{E}_{z,y}[g(z)g(z \oplus (\neg y)^\sigma)] \\ - \frac{1}{4}\mathbb{E}_{x,y,z}[f(x)f(x \oplus y^\pi)g(z)g(z \oplus (\neg y)^\sigma)] \quad (7.9)$$

We have

$$\mathbb{E}_{x,y}[f(x)f(x \oplus y^\pi)] = \sum_\alpha \hat{f}(\alpha)^2 \mathbb{E}_y[\chi_\alpha(y^\pi)] = \sum_\alpha \hat{f}(\alpha)^2 \mathbb{E}_y[\chi_{\pi[\alpha]}(y)] = 0 \qquad (7.10)$$

where the last equality holds since $|\alpha|$ is odd, and consequently $\pi[\alpha] \neq \emptyset$. Similarly, $\mathbb{E}_{x,z}[g(z)g(z\oplus$

[4] We note that $\pi[\alpha] = \sigma[\beta]$ is equivalent to $\chi_\alpha(x_{\pi(1)}, \ldots, x_{\pi(n)}) = \chi_\beta(x_{\sigma(1)}, \ldots, x_{\sigma(n)})$ for all $x \in \{-1, 1\}^m$.

$(\neg y)^\sigma)]$ vanishes. Thus the probability that the test accepts is

$$\begin{aligned}\frac{3}{4}-\frac{1}{4}\mathbb{E}_{x,y,z}f(x)f(x\oplus y^\pi)g(z)g(z\oplus(\neg y)^\sigma) \qquad (7.11)\\ =\frac{3}{4}-\frac{1}{4}\sum_{\alpha,\beta}\hat{f}(\alpha)^2\hat{g}(\beta)^2\mathbb{E}_y[\chi_\alpha(y^\pi)\chi_\beta(-y^\sigma)]\\ =\frac{3}{4}+\frac{1}{4}\sum_{\alpha,\beta}\hat{f}(\alpha)^2\hat{g}(\beta)^2\mathbb{E}_y[\chi_\alpha(y^\pi)\chi_\beta(y^\sigma)]\\ =\frac{3}{4}+\frac{1}{4}\sum_{\alpha,\beta}\hat{f}(\alpha)^2\hat{g}(\beta)^2\mathbb{E}_y[\chi_{\pi[\alpha]}(y)\chi_{\sigma[\beta]}(y)]\\ =\frac{3}{4}+\frac{1}{4}\sum_{\alpha,\beta:\pi[\alpha]=\sigma[\beta]}\hat{f}(\alpha)^2\hat{g}(\beta)^2\\ =1-\frac{1}{4}\sum_{\alpha,\beta:\pi[\alpha]\neq\sigma[\beta]}\hat{f}(\alpha)^2\hat{g}(\beta)^2\end{aligned}$$

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

7.3.3 The full test

Putting the analysis of the two tests together we get the following.

Lemma 115. *Given that both f and g are folded, the probability that the joint test accepts is at most*

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

Proof. Follows directly from Lemmas 112 and 114. □

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

Proof of Theorem 111. The completeness follows in a straightforward way from the two comments after the description of the tests. We prove the soundness. Suppose that the test passes with probability $1-\delta$. We will show that this implies that there is an assignment to the Label Cover instance that satisfies $(1-16\delta)$-fraction of constraints.

Our decoding is as follows: for a node $v\in V$, decode v to $i\in\Sigma$ with probability proportional to $\hat{f}_v(i)^2$. Intuitively, we decode to the value i with higher probability if f is closer to the code word $p_i=\chi_{\{i\}}$ or its negative $-p_i$ (see also Section 7.2.2). We will show that in expectation, this decoding satisfies at least $1-16\delta$ fraction of constraints, which proves that there exists a labeling that satisfies at least $1-16\delta$ fraction of constraints.

Let $1-\delta_e$ denote the probability that the test passes when we pick edge e. As test passes with probability $1-\delta$, we know that $\mathbb{E}_e[\delta_e]=\delta$. Suppose that we pick $e=(u,v)$ with f and g being the functions corresponding to u and v respectively. From the above lemma, we have that

$$1-\delta_e \leq 1-\frac{1}{16}\Big(\sum_{|\alpha|>1}\hat{f}(\alpha)^2+\sum_{|\beta|>1}\hat{g}(\beta)^2+\sum_{i,j:\pi(i)\neq\sigma(j)}\hat{f}(i)^2\hat{g}(j)^2\Big), \tag{7.12}$$

and therefore,

$$16\delta_e \geq \sum_{|\alpha|>1}\hat{f}(\alpha)^2+\sum_{|\beta|>1}\hat{g}(\beta)^2+\sum_{i,j:\pi(i)\neq\sigma(j)}\hat{f}(i)^2\hat{g}(j)^2. \tag{7.13}$$

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

$$\begin{aligned}
\sum_{i,j:\pi(i)=\sigma(j)}\hat{f}(i)^2\hat{g}(j)^2 &= 1-\sum_{\substack{\alpha,\beta\\ |\alpha|>1,\text{ or }|\beta|>1,\text{ or }\alpha=\{i\}\text{ and }\beta=\{j\}\text{ and }\pi(i)\neq\sigma(j)}}\hat{f}(\alpha)^2\hat{g}(\beta)^2 \qquad (7.14)\\
&\geq 1-\sum_{\substack{\alpha,\beta\\|\alpha|>1}}\hat{f}(\alpha)^2\hat{g}(\beta)^2-\sum_{\substack{\alpha,\beta\\|\beta|>1}}\hat{f}(\alpha)^2\hat{g}(\beta)^2-\sum_{\substack{i,j\\ \pi(i)\neq\sigma(j)}}\hat{f}(i)^2\hat{g}(j)^2\\
&= 1-\sum_{|\alpha|>1}\hat{f}(\alpha)^2-\sum_{|\alpha|>1}\hat{g}(\alpha)^2-\sum_{i,j:\pi(i)\neq\sigma(j)}\hat{f}(i)^2\hat{g}(j)^2\\
&\geq 1-16\delta_\epsilon
\end{aligned}$$

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

Theorem 107 is stated without the assumption that the constraints are rectangular. This slightly more general version follows from Theorem 111 by a standard reduction which we describe below, in the proof of Theorem 116.

7.4 CSP to Label Cover

In this section, we recall the basic structure of Dinur's proof of PCP Theorem, and show how the previous reduction can be used in the alphabet reduction step of Dinur's proof. The resulting proof requires a gap amplification step for which we refer to Dinur's paper [Din07].

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

Theorem 116 (Alphabet reduction). *Given a Label Cover instance $G=(V,E,\Pi,\Sigma_0)$ with rectangular constraints, there is a polynomial time reduction that outputs another Label Cover instance with rectangular constraints $G'=(V',E',\Pi',\Sigma)$ with alphabet size Σ such that $|\Sigma|$ is an absolute constant, $size(G')=O(size(G))$ and*

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

Proof. We first convert the Label Cover instance G to a CSP instance I as in Theorem 111. The CSP instance can be converted to bipartite Label Cover using standard clause-variable Label-coverization technique. We include the proof here for the sake of completeness. We have n vertices $x_1, x_2, \ldots, x_n$ corresponding to the variables of I on the left L, and m vertices corresponding to constraints $C_1, C_2, \ldots, C_m$ of I on the right R. The label set is binary on the left, and satisfying assignments (at most 16) on the right corresponding to the possible assignments to four variables in the constraint. We add an edge between $u \in L$ and $v \in R$ if $x_u \in C_v$. The constraint on this edge enforces that the assignment to x_u is consistent with the assignment C_v assigns to x_u.

If there is a satisfying labeling to G, there is a satisfying assignment to I. Using this, we can assign the variables on the left the satisfying assignment, and the corresponding assignment to tuples for the vertices of constraints on the right, and thus get a satisfying assignment to G'. Suppose that every labeling violates at least δ fraction of constraints of G. From Theorem 111, every assignment violates at least $C\delta$ fraction of constraints in I. Suppose there is a labeling to G' that satisfies δ' fraction of constraints. Consider the assignment obtained by this labeling on the left. This assignment violates at least $C\delta m$ number of constraints in I. Note that this should violate at least $C\delta m$ constraints in G' and thus $\delta' \geq C'\delta$ for an absolute constant C'. The constraints are in fact projections, and thus are rectangular too. $\square$

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

Definition 117 (Constraint Graph Powering). *Given a d-regular Label Cover (a.k.a. Constraint graph) $G = (V, E, \Sigma, \Pi)$, we obtain t-th power of it $G^t = (V, E', \Sigma', \Pi')$ as follows:*

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

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

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

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

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

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

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

Theorem 118 (Gap amplification)**.** *Fix a parameter t. Given a Label Cover $G = (V, E, \Pi, \Sigma)$ where Σ is an absolute constant, there is a polynomial time reduction to output a rectangular Label Cover instance $G' = (V', E', \Pi', \Sigma')$ with the alphabet size $|\Sigma'| = c(|\Sigma|, t)$ such that*

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

Choosing t large enough constant and iterating Theorem 116 and Theorem 118 $\log(m)$ times proves the PCP theorem.

7.5 Derandomization of the gadget decoding

In this appendix, we provide a derandomization of the decoding used in Theorem 111. This requires only a little additional argument. The idea is, instead of decoding to i with probability $\hat{f}(i)^2$, to decode to the i with the largest $\hat{f}(i)^2$. We set i_f to be such i. In this light, the reduction can be analyzed by analyzing completeness and soundness of the gadget separately without considering the rest of the instance. The following lemma then shows that the gadget has perfect completeness and soundness 99% not depending on the parameters n and m (the alphabet sizes), π and σ.

Lemma 119. *There is a gadget with inputs n, m, k, $\pi\colon [n] \to [k]$, and $\sigma\colon [m] \to [k]$ that produces an instance of $\mathrm{CSP}(R)$ with variables $f(a_1, \ldots, a_n)$ and $g(a_1, \ldots, a_m)$ such that*

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

Proof. First, we bound the probability that the test accepts. For the long code test, starting with the first expression in (7.7), we obtain the following bound.

$$
\begin{aligned}
&1 - \frac{1}{2}\sum_{|\alpha|>1} \hat{f}(\alpha)^2\,(3/8) - \frac{1}{4}\sum_{\alpha\cap\beta=\emptyset} \hat{f}(\alpha)^2\hat{f}(\beta)^2(1/2)^{|\alpha\cup\beta|} \\
\leq &1 - \frac{3}{16}\sum_{|\alpha|>1} \hat{f}(\alpha)^2 - \frac{1}{16}\sum_{i\neq j} \hat{f}(i)^2\hat{f}(j)^2
\end{aligned}
$$

For the consistency test, we use the bound from Lemma 114. Thus the overall probability that the whole test accepts is at most

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

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

$$\sum_{|\alpha|>1}\hat{f}(\alpha)^2 \leq 1/6 \tag{7.15}$$

$$\sum_{i\neq j}\hat{f}(i)^2\hat{f}(j)^2 \leq 1/2 \tag{7.16}$$

$$\sum_{|\alpha|>1}\hat{g}(\alpha)^2 \leq 1/6 \tag{7.17}$$

$$\sum_{i\neq j}\hat{g}(i)^2\hat{g}(j)^2 \leq 1/2 \tag{7.18}$$

$$\sum_{i,j:\pi(i)\neq\sigma(j)}\hat{f}(i)^2\hat{g}(j)^2 \leq 1/8 \tag{7.19}$$

From Parseval's identity and (7.15), we get that $1 \geq \sum_i \hat{f}(i)^2 \geq 5/6$. Recall that i_f is such i that $\hat{f}(i)^2$ is maximal. Then using the above and (7.16), we obtain that

$$\hat{f}(i_f)^2 \geq \hat{f}(i_f)^2\sum_i \hat{f}(i)^2 \geq \sum_i \hat{f}(i)^4 = \sum_{i,j}\hat{f}(i)^2\hat{f}(j)^2 - \sum_{i\neq j}\hat{f}(i)^2\hat{f}(j)^2 \geq (5/6)^2 - 1/2 = 4/9. \tag{7.20}$$

Similarly, from (7.17) and (7.18), we get $\hat{g}(i_g)^2 \geq 4/9$. Finally, since $\hat{f}(i_f)^2\hat{g}(i_g)^2 \geq (4/9)^2 > 1/8$, we have that $\pi(i_f) = \sigma(i_g)$ otherwise (7.19) cannot be true. $\square$

Theorem 111 can be also directly obtained from this lemma albeit with a worse constant than in the above proof: Let $C = 1\%$ and assume that $\delta < 1$. Given that the resulting CSP instance has an assignment fails no more than $C\delta$-fraction of the constraints, we derive that in at least $(1-\delta)$-fraction of the gadgets, no more than C-fraction of constraints are unsatisfied. Lemma 119 then shows that the assignment $s\colon u \mapsto i_{f_u}$ is an assignment of the Label Cover instance that satisfies all the constraints corresponding to these gadgets. This completes the proof.

Part II

Structured instances

Chapter 8

Multidimensional Packing and Scheduling Problems

8.1 Introduction

Bin Packing and Multiprocessor Scheduling (also known as Makespan Minimization) are some of the most fundamental problems in Combinatorial Optimization. They have been studied intensely from the early days of approximation algorithms and have had a great impact on the field. These two are packing problems where we have n jobs with certain sizes, and the objective is to pack them into bins efficiently. In Bin Packing, each bin has unit size and the objective is to minimize the number of bins, while in Multiprocessor Scheduling, we are given a fixed number of bins and the objective is to minimize the maximum load in a bin. These problems are well understood in terms of approximation algorithms: both the problems are NP-hard, and have a Polynomial Time Approximation Scheme (PTAS) [VL81; HS87].

In this chapter, we study the approximability of the multidimensional generalizations of these problems. The corresponding problems are Vector Bin Packing and Vector Scheduling. Apart from their theoretical importance, these problems are widely applicable in practice [Spi94; ST12; Pan+11] where the jobs often have multiple dimensions such as CPU, Hard disk, memory, etc.

In the Vector Bin Packing problem, the input is a set of n vectors in $[0, 1]^d$ and the goal is to partition the vectors into the minimum number of parts such that in each part, the sum of vectors is at most 1 in every coordinate. The problem behaves differently from Bin Packing even when $d = 2$: Woeginger [Woe97] proved[1] that there is no asymptotic[2] PTAS for 2-dimensional Vector Bin Packing, assuming $\mathsf{P} \neq \mathsf{NP}$. On the algorithmic front, the PTAS for Bin Packing [VL81] easily implies a $d + \epsilon$ approximation for Vector Bin Packing. When d is part of the input, this

[1] Very recently, Ray [Ray21] found an oversight in Woeginger's original proof and gave a revised APX hardness proof for the problem.

[2] The asymptotic approximation ratio (formally defined in Section 8.2) of an algorithm is the ratio of its cost and the optimal cost when the optimal cost is large enough. All the approximation factors mentioned in this chapter for Vector Bin Packing are asymptotic.

is almost tight: there is a lower bound of $d^{1-\epsilon}$ shown by [CK04][3]. When d is a fixed constant[4], much better algorithms are known [CK04; BCS09; BEK16] that get $\ln d + O(1)$ approximation guarantee. However, the best hardness factor (for arbitrary constant d) is still the APX-hardness result of the 2-dimensional problem due to Woeginger from 1997.

Closing this gap, either by obtaining a $O(1)$ factor algorithm or showing a hardness factor that is a function of d, has remained a challenging open problem. It is one of the ten open problems in a recent survey on multidimensional scheduling problems [Chr+17]. It also appeared in a recent report by Bansal [Ban17] on open problems in scheduling. In fact, to the best of our knowledge, super constant integrality gap instances for the configuration LP relaxation of the problem were also not known. For the integer instances i.e. when the vectors are from $\{0,1\}^d$ (which are the hard instances for Vector Scheduling and Vector Bin Covering), there is an asymptotic PTAS since there are $O_d(1)$ item types.

In the Vector Scheduling problem, given a set of n vector jobs in $[0,1]^d$, and m identical machines, the objective is to assign the jobs to machines to minimize the maximum ℓ_∞ norm of the load on the machines. Chekuri and Khanna [CK04] introduced the problem as a natural generalization of Multiprocessor Scheduling and obtained a PTAS for the problem when d is a fixed constant. When d is part of the input, they obtained a $O(\log^2 d)$ factor approximation algorithm. They also showed that it is NP-hard to obtain a C factor approximation algorithm for the problem, for any constant C. Meyerson, Roytman, and Tagiku [MRT13] gave an improved $O(\log d)$ factor algorithm while the current best factor is $O\left(\frac{\log d}{\log \log d}\right)$ due to Harris and Srinivasan [HS19] and Im, Kell, Kulkarni, and Panigrahi [Im+19]. The algorithm of Harris and Srinivasan [HS19] works for the more general setting of unrelated machines where each job can have a different vector load for each machine. However, no super constant hardness is known even in this unrelated machines setting.

8.1.1 Our Results

We prove almost optimal hardness results for both the multidimensional problems discussed above.

Vector Bin Packing

For the Vector Bin Packing problem, we prove a $\Omega(\log d)$ asymptotic hardness of approximation when d is a large constant, matching the $\ln d + O(1)$ approximation algorithms [CK04; BCS09; BEK16], up to constants.

Theorem 120. *There exists an integer d_0 and a constant $c > 0$ such that for all constants $d \geq d_0$, d-dimensional Vector Bin Packing has no asymptotic $c \log d$ factor polynomial time approximation algorithm unless $\mathsf{P} = \mathsf{NP}$.*

[3][CK04] actually give $d^{\frac{1}{2}-\epsilon}$ hardness, but it has been shown later (see e.g., [Chr+17]) that a slight modification of their reduction gives $d^{1-\epsilon}$ hardness.

[4]The algorithms are now allowed to run in time $n^{f(d)}$, for some function f.

We obtain our hardness result via a reduction from the set cover problem on certain structured instances. In the set cover problem, we are given a set system $\mathcal{S} \subseteq 2^V$ on a universe V, and the goal is to pick the minimum number of sets from $\mathcal{S}$ whose union is V. Observe that Vector Bin Packing is a special case of the set cover problem with the vectors being the elements and every maximal set of vectors whose sum is at most 1 in every coordinate (known as "configurations") being the sets. In fact, in the elegant Round & Approx framework [BCS09; BEK16], the Vector Bin Packing problem is viewed as a set cover instance, and the algorithms proceed by rounding the standard set cover LP. Towards proving the hardness of Vector Bin Packing, we ask the converse: *Which families of set cover instances can be cast as d-dimensional Vector Bin Packing?*

We formalize this question using the notion of *packing dimension* of a set system $\mathcal{S}$ on a universe V: it is the smallest integer d such that there is an embedding $f : V \to [0, 1]^d$ such that a set $S \subseteq V$ is in $\mathcal{S}$ if and only if

$$\left\| \sum_{v \in S} f(v) \right\|_\infty \leq 1$$

If a set system has packing dimension d, then the corresponding set cover problem can be embedded as a d-dimensional Vector Bin Packing instance. However, it is not clear if the hard instances of the set cover problem have a low packing dimension. Indeed the instances in the $(1 - \epsilon) \ln n$ set cover hardness [Fei98] have a large packing dimension that grows with n, which we cannot afford as we are operating in the constant d regime. We get around this by starting our reduction from highly structured yet hard instances of set cover. In particular, we study simple bounded set systems which satisfy the following three properties:

1. The set system is *simple*[5] i.e., every pair of sets intersect in at most one element.
2. The cardinality of each set is at most k, a fixed constant.
3. Each element in the family is present in at most $\Delta = k^{O(1)}$ sets.

Kumar, Arya, and Ramesh [KAR00] proved that simple set cover i.e., set cover with the restriction that every pair of sets intersect in at most one element, is hard to approximate within $\Omega(\log n)$. We observe that by modifying the parameters slightly in their proof, we can obtain the $\Omega(\log k)$ hardness of simple bounded set cover.

We prove that simple bounded set systems have packing dimension at most $k^{O(1)}$. Thus, the $\Omega(\log k)$ simple bounded set cover hardness translates to $\Omega(\log d)$ hardness of Vector Bin Packing when d is a constant. Note that the optimal value of the set cover instances can be made arbitrarily large in terms of k by starting with a Label Cover instance with an arbitrarily large number of edges. Thus, our Vector Bin Packing hardness holds for asymptotic approximation as well.

Our upper bound on the packing dimension is obtained in two steps: First, we write the given simple bounded set system as an intersection of $(k\Delta)^{O(1)}$ structured simple bounded set systems on the same universe, and then we give an embedding using $(k\Delta)^{O(1)}$ dimensions bounding the packing dimension of these structured simple bounded set systems. This idea of decomposition into structured instances is inspired from a work of Chandran, Francis, and Sivadasan [CFS08] where an upper bound on the Boxicity of a graph is obtained in terms of its maximum degree. We

[5]Simple set families are also known as linear set families.

believe that the packing dimension of set systems is worth studying on its own, especially in light of its close connections to the notions of dimension of graphs such as Boxicity.

Vector Scheduling

For the Vector Scheduling problem, we obtain a $\Omega\left((\log d)^{1-\epsilon}\right)$ hardness under $\mathsf{NP} \not\subseteq \mathsf{ZPTIME}\left(n^{(\log n)^{O(1)}}\right)$, almost matching the $O\left(\frac{\log d}{\log\log d}\right)$ algorithms [HS19; Im+19].

Theorem 121. *For every constant $\epsilon > 0$, assuming $\mathsf{NP} \not\subseteq \mathsf{ZPTIME}\left(n^{(\log n)^{O(1)}}\right)$, d-dimensional Vector Scheduling has no polynomial time $\Omega\left((\log d)^{1-\epsilon}\right)$-factor approximation algorithm when d is part of the input.*

We obtain the hardness result via a reduction from the Monochromatic Clique problem. In the Monochromatic-Clique(k,B) problem, given a graph $G = (V, E)$ with $|V| = n$ and parameters $k(n)$ and $B(n)$, the goal is to distinguish between the case when G is k-colorable and the case when in any assignment of k-colors to vertices of G, there is a clique of size B all of whose vertices are assigned the same color. When $B = 2$, this is the standard graph coloring problem. Note that the problem gets easier as B increases. Indeed, when $B > \sqrt{n}$, we can solve the problem in polynomial time by computing the Lovász theta function of the complement graph. We are interested in proving the hardness of the problem for B as large a function of n as possible, for some k. For example, given a graph that is promised to be k colorable, can we prove the hardness of assigning k colors to the vertices of the graph in polynomial time where each color class has maximum clique at most $B = \log n$?

The Monochromatic Clique problem was defined formally by Im, Kell, Kulkarni, and Panigrahi [Im+19] in the context of proving lower bounds for online Vector Scheduling. It was also used implicitly in the $\omega(1)$ NP-hardness of Vector Scheduling by Chekuri and Khanna [CK04]. They proved (implicitly) that Monochromatic Clique is NP-hard when B is an arbitrary constant using a reduction from $n^{1-\epsilon}$ hardness of graph coloring. We observe that the same reduction combined with better hardness of graph chromatic number [Kho01] proves the hardness of Monochromatic Clique when $B = (\log n)^{\gamma}$, for some constant $\gamma > 0$ under the assumption that $\mathsf{NP} \not\subseteq \mathsf{ZPTIME}\left(n^{(\log n)^{O(1)}}\right)$.

We then amplify this hardness to $B = (\log n)^{C}$, for *every* constant $C > 0$. Our main idea in this amplification procedure is the notion of a stronger form of Monochromatic Clique where given a graph and parameters k, B, C, the goal is to distinguish between the case that G is k colorable vs. in any k^{C} coloring of G, there is a monochromatic clique of size B. It turns out that the graph coloring hardness of Khot [Kho01] already proves the hardness of this stronger variant of Monochromatic clique when $B = (\log n)^{\gamma}$ for any constant C. We then use lexicographic product of graphs to amplify this result into the hardness of original Monochromatic Clique problem with $B = (\log n)^{C}$ for any constant C under the same assumption that $\mathsf{NP} \not\subseteq \mathsf{ZPTIME}\left(n^{(\log n)^{O(1)}}\right)$. This directly gives the required hardness of Vector Scheduling using the reduction in [CK04].

The Vector Scheduling problem is also closely related to the Balanced Hypergraph Coloring problem where the input is a hypergraph H and a parameter k, and the objective is to color the vertices of H using k colors to minimize the maximum number of times a color appears in an edge. We use this connection to improve upon the NP-hardness of the problem:

Theorem 122. *For every constant $C > 0$, d-dimensional Vector Scheduling is NP-hard to approximate within $\Omega\left((\log \log d)^{C}\right)$ when d is part of the input.*

Consider the case when each vector job is from $\{0,1\}^d$. In this setting, we can view each coordinate as an edge in a hypergraph, and each vector corresponds to a vertex of the hypergraph. The goal is to find a m-coloring of vertices of the hypergraph i.e., an assignment of the vectors to m machines to minimize the maximum number of monochromatic vertices in an edge, which directly corresponds to the maximum load on a machine.

This problem of coloring a hypergraph to ensure that no color appears too many times in each edge is known as Balanced Hypergraph Coloring. Guruswami and Lee [GL18] obtained strong hardness results for this problem when k, the uniformity of the hypergraph is a constant, using the Label Cover Long Code framework combined with analytical techniques such as the invariance principle. However, when k is super constant, the invariance principle based methods give weak bounds as the soundness of the Label Cover has to be at least exponentially small in k. Recently, using combinatorial tools to analyze the gadgets instead of the standard analytical techniques, improvements have been obtained for various hypergraph coloring problems [Bha18; ABP19] in the super-constant inapproximability regime. We follow the same route and use combinatorial tools to analyze the gadgets in the Label Cover Long Code framework and obtain better hardness of Balanced Hypergraph Coloring in the regime of super-constant uniformity k. The key combinatorial lemma used in our analysis was proved recently by Austrin, Bhangale, and Potukuchi [ABP20] using a generalization of the Borsuk-Ulam theorem.

The NP-hardness of Vector Scheduling follows directly from the hardness of Balanced Hypergraph Coloring using the above-described reduction. This NP-hardness result uses near-linear size Label Cover hardness results [MR10; DS14]. By using the standard Label Cover hardness obtained by combining PCP Theorem and Parallel Repetition in the same reduction, we also prove an intermediate result bridging the above two hardness results for Vector Scheduling.

Theorem 123. *There exists a constant $\gamma > 0$ such that assuming $\mathsf{NP} \nsubseteq \mathsf{DTIME}\left(n^{O(\log \log n)}\right)$, d-dimensional Vector Scheduling is hard to approximate within $\Omega\left((\log d)^{\gamma}\right)$ when d is part of the input.*

8.1.2 Related Work

Online Algorithms. Multidimensional packing problems have been extensively studied in the online setting. For the d-dimensional Vector Bin Packing, the classical First-Fit algorithm [Gar+76] gives $O(d)$ competitive ratio, and Azar, Cohen, Kamara, and Shepherd [Aza+13] recently gave an almost matching $\Omega\left(d^{1-\epsilon}\right)$ lower bound. For the d-dimensional Vector Scheduling, Im, Kell, Kulkarni, and Panigrahi [Im+19] gave a $O\left(\frac{\log d}{\log \log d}\right)$ competitive online algorithm and proved a matching lower bound.

Geometric variants. There are various natural geometric variants of Vector Bin Packing that have been studied in the literature. A classical problem of this sort is the 2-dimensional Geometric Bin

Packing, where the input is a set of rectangles that need to be packed into the minimum number of unit squares. After a long line of works, Bansal and Khan [BK14a] gave a 1.405 factor asymptotic approximation algorithm for the problem. On the hardness front, Bansal and Sviridenko [BS04] showed that the problem does not admit an asymptotic PTAS, and this APX hardness result has been generalized to several related problems by Chlebík and Chlebíková [CC06]. We refer the reader to the excellent survey [Chr+17] regarding the geometric problems.

8.1.3 Organization

We first define the multidimensional problems and the Label Cover problem formally in Section 8.2. Next, we prove the hardness results for Vector Bin Packing and Vector Scheduling in Section 8.3, Section 8.4 respectively.

8.2 Preliminaries

Notations. We use $[n]$ to denote the set $\{1, 2, \dots, n\}$. We use $\mathbf{1}^d$ to denote the d-dimensional vector $(1, 1, \dots, 1)$. For two d-dimensional real vectors $\mathbf{a}$ and $\mathbf{b}$, we say that $\mathbf{a} \geq \mathbf{b}$ if $\mathbf{a}_i \geq \mathbf{b}_i$ for all $i \in [d]$. For a graph G, we let $\omega(G), \alpha(G), \chi(G)$ be the largest clique size, largest independent size, and the chromatic number of G respectively. A set system or set family $\mathcal{S}$ on a universe V is a collection of subsets of V.

Problem Statements. We give formal definitions for the problems that we study.

Definition 124. *(Vector Bin Packing) In the Vector Bin Packing problem, the input is a set of n rational vectors $v_1, v_2, \dots, v_n \in [0, 1]^d$. The objective is to partition $[n]$ into minimum number of parts $A_1, A_2, \dots, A_m$ such that*

$$\Big\| \sum_{j \in A_i} v_j \Big\|_\infty \leq 1 \ \forall i \in [m]$$

Definition 125. *(Vector Scheduling) In the Vector Scheduling problem, the input is a set of n rational vector jobs $v_1, v_2, \dots, v_n \in [0, 1]^d$, and m identical machines. The objective is to assign the jobs to machines i.e. partition $[n]$ into m parts $A_1, A_2, \dots, A_m$ to minimize the makespan which is defined as the maximum ℓ_∞ load on a machine.*

$$\max_{i \in [m]} \Big\| \sum_{j \in A_i} v_j \Big\|_\infty$$

Asymptotic Approximation. For the Bin Packing problem, it is NP-Hard to identify if all the vectors can be packed into 2 bins or need 3 bins. This already proves that the problem is NP-hard to approximate within $\frac{3}{2}$ as per the usual notion of multiplicative approximation ratio. However, this is less interesting as there are much better *asymptotic* approximation algorithms for the problem which get $(1 + \epsilon)$-factor approximation when the optimal value is large enough, for every positive constant $\epsilon > 0$.

Even for the Vector Bin Packing problem, the performance of an algorithm is typically measured in the asymptotic setting. We give the formal definition [Chr+17] of asymptotic approximation ratio of an algorithm $\mathcal{A}$ for the Vector Bin Packing problem.

Definition 126. *(Asymptotic Approximation Ratio) The asymptotic approximation ratio $\rho_{\mathcal{A}}^{\infty}$ of an algorithm $\mathcal{A}$ for the Vector Bin Packing problem is*

$$\rho_{\mathcal{A}}^{\infty} = \limsup_{n\to\infty} \rho_{\mathcal{A}}^{n}, \ \ \rho_{\mathcal{A}}^{n} = \sup_{I\in\mathcal{I}} \left\{ \frac{\mathcal{A}(I)}{\text{OPT}(I)} : \text{OPT}(I) = n \right\}$$

where $\mathcal{I}$ denotes the set of all possible Vector Bin Packing instances.

All the results mentioned in this chapter regarding Vector Bin Packing are with respect to the asymptotic approximation ratio.

Label Cover. We define the Label Cover problem:

Definition 127. *(Label Cover) In an instance of the Label Cover problem $G = (V = L \cup R, E, \Sigma_L, \Sigma_R, \Pi)$ with $|\Sigma_L| \geq |\Sigma_R|$, the input is a bipartite graph $L \cup R$ with constraints on every edge. The constraint on an edge e is a projection $\Pi_e : \Sigma_L \to \Sigma_R$. We say a labeling $\sigma : V \to \Sigma_L \cup \Sigma_R$ satisfies the constraint on the edge $e = (u, v)$ if $\Pi_e(\sigma(u)) = \sigma(v)$. The objective is to find a labeling $\sigma : V \to \Sigma_L \cup \Sigma_R$ that satisfies as many constraints as possible.*

By a simple reduction from the 3-SAT problem, we can prove that Label Cover is NP-hard when Σ_L and Σ_R are constants (See e.g., Lemma 4.2 in [BG16]).

Theorem 128. *Given a Label Cover instance when $\Sigma_L = \Sigma_R = [6]$, it is NP-hard to identify if it has a labeling that satisfies all the constraints.*

The real use of Label Cover, however, lies in its strong hardness of approximation. PCP Theorem [Aro+98] combined with Raz's parallel repetition [Raz98] yields the following strong inapproximability of Label Cover problem.

Theorem 129. *There exists an absolute constant $c > 1$ such that for every integer n and $\epsilon > 0$, there is a reduction from 3-SAT instance I over n variables to Label Cover instance $G = (V = L \cup R, E, \Sigma_L, \Sigma_R, \Pi)$ with $|V| \leq n^{O\left(\log\left(\frac{1}{\epsilon}\right)\right)}$, $|\Sigma_L| \leq \left(\frac{1}{\epsilon}\right)^c$ satisfying the following:*

1. *(Completeness.) If I is satisfiable, there exists a labeling to G that satisfies all the constraints.*
2. *(Soundness.) If I is not satisfiable, no labeling can satisfy an ϵ fraction of the constraints of G.*
3. *(Biregularity.) The graph $L \cup R, E$ is biregular with degrees on either side bounded by $\mathsf{poly}\left(\frac{1}{\epsilon}\right)$.*

Furthermore, the running time of the reduction is $\mathsf{poly}\left(n, \frac{1}{\epsilon}\right)$.

Moshkovitz-Raz [MR10] proved the following hardness of near linear size Label Cover.

Theorem 130. *There exist absolute constants $c, c' > 1$ such that for every n and $\epsilon > 0$, there is a reduction from 3-SAT instance I over n variables to Label Cover instance $G = (V = L \cup R, E, \Sigma_L, \Sigma_R, \Pi)$ with $|V| \leq n^{1+o(1)} \left(\frac{1}{\epsilon}\right)^c$, $|\Sigma_L| \leq 2^{\left(\frac{1}{\epsilon}\right)^{c'}}$ satisfying the following:*

1. *(Completeness.) If I is satisfiable, there exists a labeling to G that satisfies all the constraints.*

2. *(Soundness.) If I is not satisfiable, no labeling can satisfy an ϵ fraction of the constraints of G.*
3. *(Biregularity.) The graph $L \cup R, E$ is biregular with degrees on either side $\mathsf{poly}\left(\frac{1}{\epsilon}\right)$.*

Furthermore, when ϵ is a constant, the running time of the reduction is $\mathsf{poly}(n)$.

8.3 Vector Bin Packing

In this section, we prove the hardness of approximation of Vector Bin Packing. First, we define the packing dimension of a set family and bound the packing dimension of simple set families. Next, we combine this upper bound with the hardness of set cover on simple bounded set systems to prove Theorem 120.

8.3.1 Packing Dimension

For a set family $\mathcal{S}$ on a universe V, we define the packing dimension $\mathsf{pdim}(\mathcal{S})$ below. For a function $f : V \to [0,1]^K$ and a set $S \subseteq V$, we let $f(S)$ denote the vector $f(S) = \sum_{v \in S} f(v)$.

Definition 131. *For a set family $\mathcal{S}$ on a universe V, the packing dimension $\mathsf{pdim}(\mathcal{S})$ is defined as the smallest positive integer K such that there exists an embedding $f : V \to [0,1]^K$ that satisfies the following property: For every set $S \subseteq V$, S is in the family $\mathcal{S}$ if and only if*

$$\|f(S)\|_\infty \leq 1.$$

If no such embedding exists, we say that $\mathsf{pdim}(\mathcal{S})$ is infinite.

For a set family $\mathcal{S}$ to have finite packing dimension i.e. for an embedding $f : V \to [0,1]^K$ realizing the above condition to exist requires two conditions:

1. The set family is *downward closed* i.e. for every $S \in \mathcal{S}$ and $T \subseteq S$, $T \in \mathcal{S}$ as well.
2. For every element $v \in V$, there is a set $S \in \mathcal{S}$ with $v \in S$. We call a set family $\mathcal{S}$ on a universe V non-trivial if for every $v \in V$, there is a set $S \in \mathcal{S}$ with $v \in S$.

On the other hand, any set system that satisfies the above two conditions i.e. being downward closed and non-trivial has a finite packing dimension. Before proving this statement, we first prove the following simple but useful proposition.

Proposition 132. *For a pair of set families $\mathcal{S}_1$ and $\mathcal{S}_2$ defined on the same universe V such that $\mathsf{pdim}(\mathcal{S}_1)$ and $\mathsf{pdim}(\mathcal{S}_2)$ are finite,*

$$\mathsf{pdim}(\mathcal{S}_1 \cap \mathcal{S}_2) \leq \mathsf{pdim}(\mathcal{S}_1) + \mathsf{pdim}(\mathcal{S}_2)$$

Proof. Let $K_1 = \mathsf{pdim}(\mathcal{S}_1)$ and $K_2 = \mathsf{pdim}(\mathcal{S}_2)$. Suppose that $f_1 : V \to [0,1]^{K_1}$ be such that for every set $S \subseteq V$,

$$\|f_1(S)\|_\infty \leq 1$$

if and only if $S \in \mathcal{S}_1$. Similarly, let $f_2 : V \to [0,1]^{K_2}$ be such that for every set $S \subseteq V$,

$$\|f_2(S)\|_\infty \leq 1$$

if and only if $S \in \mathcal{S}_2$. Consider the function $f : V \to [0,1]^{K_1+K_2}$ defined as $f(v) = (f_1(v), f_2(v))$. Then, for every set $S \subseteq V$, $\|f(S)\|_\infty \le 1$ if and only if $\|f_1(S)\|_\infty \le 1$ and $\|f_2(S)\|_\infty \le 1$. Thus, for every set $S \subseteq V$, $\|f(S)\|_\infty \le 1$ if and only if $S \in \mathcal{S}_1$ and $S \in \mathcal{S}_2$, or equivalently, if $S \in \mathcal{S}_1 \cap \mathcal{S}_2$. Hence, the packing dimension of $\mathcal{S}_1 \cap \mathcal{S}_2$ is at most $K_1 + K_2$. $\square$

For a set $S \subseteq V$, let $S^\uparrow$ be the family of sets $T \subseteq V$ such that $S \subseteq T$. Similarly, let $S^\downarrow$ be the family of sets $T \subseteq V$ such that $T \subseteq S$. For a set system $\mathcal{S}$, we let $\mathcal{S}^\uparrow$ (resp. $\mathcal{S}^\downarrow$) denote the union of $S^\uparrow$ (resp. $S^\downarrow$) over all $S \in \mathcal{S}$.

Consider a set $S \subseteq V$ with $|S| > 1$. For the set family $2^V \setminus S^\uparrow$, we have the embedding $f : V \to [0,1]$ defined as

$$f(v) = \begin{cases} \frac{1}{|S|} + \frac{1}{|S|^2}, & \text{if } v \in S \\ 0 & \text{otherwise.} \end{cases}$$

This shows that $\mathsf{pdim}(2^V \setminus S^\uparrow) \le 1$ for all $S \subseteq V$ with $|S| > 1$. Note that we have

$$\mathcal{S} = \bigcap_{S \notin \mathcal{S}} 2^V \setminus S^\uparrow$$

for every downward closed set system $\mathcal{S}$. Combined with Proposition 132, we obtain that for every non-trivial downward closed family $\mathcal{S}$ on a universe V, $\mathsf{pdim}(\mathcal{S}) \le 2^{|V|}$.

We are interested in the classes of set families for which there is an efficient embedding with packing dimension being independent of $|V|$. In particular, the class of set families that we study are bounded set families where each set has cardinality at most k, and each element appears in at most Δ sets. We can show that such bounded set families that are downward closed and non-trivial have packing dimension at most $(k\Delta)^{O(\Delta)}$. Together with the $\Omega(\log k)$ hardness [Tre01] of k-set cover where each set has cardinality at most k, a fixed constant and each element appearing in $(\log k)^{O(1)}$ sets, this packing dimension bound gives the hardness of $(\log d)^{\Omega(1)}$ for the Vector Bin Packing problem when d is a large constant. Unfortunately, the exponential dependence on Δ is necessary for the packing dimension of bounded set systems, and thus, this approach does not yield the optimal $\Omega(\log d)$ hardness of Vector Bin Packing.

Instead of using arbitrary bounded set families, we bypass this barrier by using simple bounded set families. Recall that a set family is called simple if any two distinct sets in the family intersect in at most one element. It turns out that for simple bounded set families i.e. simple set families $\mathcal{S}$ where each set has cardinality at most k, and each element appears in at most Δ sets, the packing dimension of $\mathcal{S}^\downarrow$ can be upper bounded by $(k\Delta)^{O(1)}$. Together with the $\Omega(\log k)$ hardness of simple k-set cover (proved in Section 8.5), we get the optimal $\Omega(\log d)$ hardness of Vector Bin Packing when d is a large constant. In the next subsection, we prove the packing dimension upper bound, and we use this upper bound to prove the hardness of Vector Bin Packing in Section 8.3.3.

8.3.2 Packing Dimension of Simple Bounded Set Families

The main embedding result that we prove is that the downward closure of simple set systems where each set has cardinality k and each element appears in at most Δ sets has packing dimension at most polynomial in k, Δ.

Theorem 133. *Suppose that $\mathcal{S}$ is a simple non-trivial set system on a universe V where each set has cardinality at most $k \geq 2$ and each element appears in at most Δ sets. Then,*

$$\mathsf{pdim}(\mathcal{S}^{\downarrow}) \leq (k\Delta)^{O(1)}$$

Furthermore, an embedding realizing the above can be found in time polynomial in $|V|$.

We prove the embedding result by writing the set family $\mathcal{S}^{\downarrow}$ as an intersection of $(k\Delta)^{O(1)}$ structured set families each of which has packing dimension at most $(k\Delta)^{O(1)}$. We can then upper bound the packing dimension of $\mathcal{S}^{\downarrow}$ using Proposition 132. The structured set systems we study are *sunflower-bouquets*, which are a disjoint union of sunflowers[6] that have a single element as the kernel. The formal definition of the sunflower-bouquet set families is below. See Figure 8.1 for an illustration.

Definition 134. *(Sunflower-bouquets) A simple set system $\mathcal{S}$ on a universe V is called a sunflower-bouquet with core $U \subseteq V, U \neq \phi$ if the following hold.*

1. *Every set $S \in \mathcal{S}$ satisfies $|S \cap U| = 1$. Furthermore, for every $u \in U$, there is a set $S \in \mathcal{S}$ with $u \in S$.*
2. *For any pair of sets $S_1, S_2 \in \mathcal{S}$ with $S_1 \cap S_2 \neq \emptyset$, we have $S_1 \cap U = S_2 \cap U = S_1 \cap S_2$.*

We now give an efficient embedding for a sunflower-bouquet $\mathcal{S}$ on a universe V with core $U \subseteq V, U \neq \phi$. The motivation behind this lemma is to upper bound the packing dimension of the set system $\mathcal{T}^{\downarrow} = \mathcal{S}^{\downarrow} \cup \{S \subseteq V \setminus U : |S| \leq k\}$.

Lemma 135. *Fix an integer $k \geq 2$. Let $\mathcal{S}$ be a simple set family defined on a universe V that is a sunflower-bouquet with core U. Furthermore, each set in the family has cardinality at most k and each element appears in at most Δ sets. Then, there exists an embedding $f : V \to [0,1]^K$ that satisfies*

(A) For every set $S \in \mathcal{S}$,

$$\|f(S)\|_\infty \leq 1.$$

(B) For every set $S \notin \mathcal{S}^{\downarrow}$ with $S \cap U \neq \emptyset$,

$$\|f(S)\|_\infty > 1.$$

(C) For every set $S \subseteq V$ with $S \cap U = \emptyset$ and $|S| \leq k$,

$$\|f(S)\|_\infty \leq 1.$$

(D) For every set $S \subseteq V$ with $|S| > k$,

$$\|f(S)\|_\infty > 1.$$

with $K = (k\Delta)^{O(1)}$. Furthermore, such an embedding can be found in time polynomial in $|V|$ given $\mathcal{S}$.

[6] A sunflower is a collection of sets $S_1, S_2, \ldots, S_r$ whose pairwise intersection is constant i.e., there exists U such that $U = S_i \cap S_j$ for all $i, j \in [r], i \neq j$. This constant intersection U is called the kernel of the sunflower.

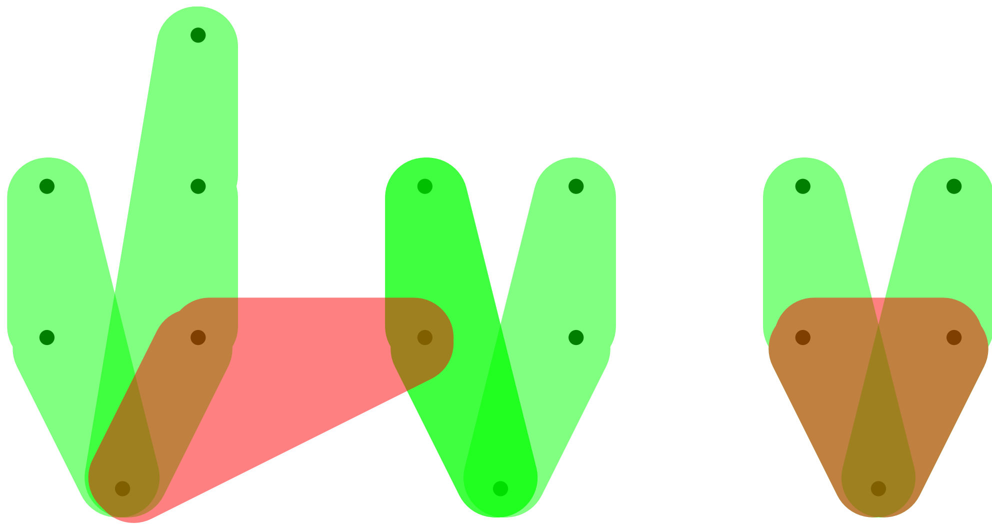

Figure 8.1: An illustration of a sunflower-bouquet set family. Here, $\mathcal{S}$ is the family of all the green colored sets. It is a sunflower-bouquet with core $U = \{u_1, u_2, u_3\}$. In the embedding, we ensure that the ℓ_∞ norm of the left red set is greater than 1 in the first step while the right side red set is handled in the second step.

Proof. Let $U = \{u_1, u_2, \dots, u_m\}$. We can partition $V \setminus U$ into $V_0, V_1, \dots, V_m$ with

$$V_i = \bigcup_{S \in \mathcal{S} : u_i \in S} S \setminus \{u_i\}$$

for all $i \in [m]$. Here, $\mathcal{S}$ restricted to $\{u_i\} \cup V_i$ is a sunflower set system with a single element u_i as the kernel for every $i \in [m]$. As each set in $\mathcal{S}$ has cardinality at most k and each element appears in at most Δ sets, we get that $|V_i| \leq k\Delta$ for all $i \in [m]$. For every $i \in [m]$, we order the elements of V_i as $\{v_{i,1}, v_{i,2}, \dots, v_{i,k\Delta}\}$ (with repetitions if needed).

We construct the final embedding f as a concatenation of embeddings with smaller dimensions $f := (f_0, g, g')$ where $f_0 : V \to [0,1]^2$, $g : V \to [0,1]^{K_1}$ and $g' : V \to [0,1]^{K_2}$ all satisfy the conditions (A) and (C). In other words, for every set $S \subseteq V$ satisfying either $S \in \mathcal{S}$ or $S \cap U = \phi$ and $|S| \leq k$, we have $\|f_0(S)\|_\infty \leq 1$, $\|g(S)\|_\infty \leq 1$, and $\|g'(S)\|_\infty \leq 1$. Note that for a set $S \subseteq V$, we have

$$\|f(S)\|_\infty = \max(\|f_0(S)\|_\infty, \|g(S)\|_\infty, \|g'(S)\|_\infty)$$

Thus, if f_0, g, g' satisfy the conditions (A) and (C), the final embedding f also satisfies the conditions (A) and (C). Furthermore, the parameters K_1 and K_2 are chosen such that the final dimension of f, $2 + K_1 + K_2$ is at most $(k\Delta)^{O(1)}$.

First, we define the embedding $f_0 : V \to [0,1]^2$ that satisfies the conditions (A) and (C) with the additional property that for every set $S \subseteq V$ with $|S| > k$, or if $S \cap U \neq \phi$ and $S \cap V_0 \neq \phi$, or if $|S \cap U| > 1$, we have $\|f_0(S)\|_\infty > 1$. We obtain this by a simple two-dimensional embedding as follows:

$$f_0(v) = \begin{cases} \left(1, \frac{1}{k}\right), & \text{if } v \in U \\ \left(\frac{1}{k}, \frac{1}{k}\right), & \text{if } v \in V_0 \\ \left(0, \frac{1}{k}\right) & \text{otherwise.} \end{cases}$$

We can verify that f_0 satisfies the conditions (A) and (C). Furthermore, suppose that $\|f_0(S)\|_\infty \leq 1$ for a set $S \subseteq V$. Then, we can obtain the following observations that we will use later.

1. As $f_0(v)_1 = 1$ for all $v \in U$, $|S \cap U| \leq 1$. As $f_0(v)_1 = \frac{1}{k}$ for all $v \in V_0$, if $S \cap U \neq \emptyset$, then $S \cap V_0 = \emptyset$.
2. As $f_0(v)_2 = \frac{1}{k}$ for all $v \in V$, $|S| \leq k$. Thus, for every set $S \subseteq V$ such that $|S| > k$, we have $\|f_0(S)\|_\infty > 1$, and hence, $\|f(S)\|_\infty > 1$. This already proves the condition (D) of the lemma.

Overview of rest of the proof. We now restrict our attention to sets $S \subseteq V$ such that $|S| \leq k, S \cap V_0 = \phi$, and $|S \cap U| \leq 1$. Our goal is to find an embedding for these sets that satisfies the conditions (A), (B), and (C). This is the technically challenging part of the proof and requires setting various coordinates carefully to encode the properties of the set system. We break this down into two steps: eliminating the "cross-sunflower" sets, and pinning down the "intra-sunflower" sets. We give an overview of the ideas used in the two steps before presenting the full proof.

1. The cross-sunflower sets are the sets $S \subseteq V$ that contain u_i for some $i \in [m]$, but also intersect another sunflower i.e., $S \cap V_{i'} \neq \phi$ for an $i' \neq i$. Note that such a set S satisfies $S \notin \mathcal{S}^\downarrow$ and $S \cap U \neq \phi$, and thus, to satisfy the condition (B), we need to ensure that $\|f(S)\|_\infty > 1$ for such sets. We achieve this by constructing an embedding $g : V \to [0,1]^{K_1}$ that satisfies the conditions (A) and (C) and has $\|g(S)\|_\infty > 1$ for all cross-sunflower sets. We illustrate the idea used in constructing this embedding using a toy example. Suppose that we have a set of pairs of elements $(u_1, v_1), (u_2, v_2), \ldots, (u_n, v_n)$, and let their union be denoted by $W = \{u_1, v_1, u_2, v_2, \ldots, u_n, v_n\}$. Our goal is to find an embedding $g : W \to [0,1]^2$ such that
   (a) $\|g(u_i) + g(v_i)\|_\infty \leq 1$ for all $i \in [n]$.
   (b) $\|g(u_i) + g(v_j)\|_\infty > 1$ for all $i, j \in [n], i \neq j$.
   We construct this embedding by choosing n distinct real numbers $\alpha_1, \alpha_2, \ldots, \alpha_n \in (0,1)$ and setting $g(u_i) = (\alpha_i, 1 - \alpha_i)$ and $g(v_i) = (1 - \alpha_i, \alpha_i)$ for all $i \in [m]$. Note that $\|g(u_i) + g(v_j)\|_1 = 2$ for all i, j, and thus, $\|g(u_i) + g(v_j)\|_\infty = 1$ if and only if $g(u_i) + g(v_j) = (1,1)$, or equivalently, when $i = j$. The actual construction extends this idea with two differences: first, the V_is could have more than one element, and we use the pairs idea multiple times to account for this, and second, we need to ensure that the sum of the embedding of any k elements in $V \setminus (U \cup V_0)$ has ℓ_∞ norm at most 1, and thus, we need to choose the embedding of u_i to be $(\alpha_i, 2 - \frac{1}{k} - \alpha_i)$ and set $\alpha_i \in (1 - \frac{1}{k}, 1)$.
2. The intra-sunflower sets are the sets $S \subseteq V$ such that $u_i \in S$ and $S \subseteq V_i \cup \{u_i\}$ for some $i \in [m]$. We need to ensure that every intra-sunflower set S such that $S \notin \mathcal{S}^\downarrow$ satisfies $\|f(S)\|_\infty > 1$. We achieve this by constructing an embedding $g' : V \to [0,1]^{K_2}$ that

satisfies the conditions (A), (C) and has $\|g'(S)\|_\infty > 1$ for every intra-sunflower set S such that $S \notin \mathcal{S}^\downarrow$.

Fix an $i \in [m]$. For every intra-sunflower set $S \subseteq \{u_i\} \cup V_i$ with $u_i \in S$ and $S \notin \mathcal{S}^\downarrow$, we use a single dimensional embedding $g_S : V \to [0,1]$ that satisfies the conditions (A) and (C) but has $\|g_S(S)\|_\infty > 1$. We achieve this by setting $g_S(u_i) = 1 - \frac{|S|-1}{k} + \epsilon$, and $g_S(v) = \frac{1}{k}$ for every $v \in V_i \cap S$, and $g_S(v) = 0$ for all $v \in V \setminus S$, where $\epsilon < \frac{1}{k}$ is a small positive constant. Note that $\|g_S(T)\| \leq 1$ for every set T such that $T \not\supseteq S$, and since $g_S(v) \leq \frac{1}{k}$ for all $v \in V \setminus U$, the embedding g_S satisfies the conditions (A) and (C).

We can construct such a single dimensional embedding for every intra-sunflower set $S \notin \mathcal{S}^\downarrow$ and take their concatenation to obtain the required embedding g'. However, there could be exponential (in k, Δ) number of such intra-sunflower sets $S \subseteq \{u_i\} \cup V_i$, $u_i \in S$ such that $S \notin \mathcal{S}^\downarrow$. We get around this issue by observing that we need the single dimensional embedding only for *minimal* intra-sunflower sets that don't belong to $\mathcal{S}^\downarrow$. In fact, as the set system $\mathcal{S}^\downarrow$ restricted to $\{u_i\} \cup V_i$ is a sunflower with single element u_i as the kernel, we can deduce that every minimal intra-sunflower set S with $S \notin \mathcal{S}^\downarrow$ is of the form $\{u_i, x, y\}$ where $x, y \in V_i$. Thus, we can construct the single dimensional embedding for such sets and take their concatenation to obtain the required embedding g' with dimension at most $|V_i|^2 \leq (k\Delta)^2$.

As every set $S \subseteq V$ such that $S \cap V_0 = \phi$, $|S \cap U| = 1$ is either a cross-sunflower set or an intra-sunflower set, the two steps together prove that $f = (f_0, g, g')$ satisfies the conditions (A), (B) and (C).

We now present the full formal proof of the two steps.

Step-1. Eliminating the cross-sunflower sets. In the first step, our goal is to find an embedding $g : V \to [0,1]^{K_1}$ with $K_1 = 2k\Delta$ such that

1. g satisfies the conditions (A) and (C) i.e. for every set $S \in \mathcal{S}$, $\|g(S)\|_\infty \leq 1$, and for every set $S \subseteq V$ with $S \cap U = \phi$ and $|S| \leq k$, $\|g(S)\|_\infty \leq 1$.
2. For every "cross-sunflower" set $S \subseteq V$ with $u_i \in S$ for some $i \in [m]$, and $S \cap V_{i'} \neq \phi$ for $i' \in [m], i' \neq i$, we have $\|g(S)\|_\infty > 1$.

We achieve this by setting $g = (f_1, \ldots, f_{k\Delta})$, where each $f_l : V \to [0,1]^2, l \in [k\Delta]$ satisfies the conditions (A) and (C), and overall, the embedding g satisfies the second condition above.

We choose m distinct rational numbers $\alpha_1, \ldots, \alpha_m$ with $1 - \frac{1}{k} < \alpha_i < 1$ for all $i \in [m]$. We define the embeddings $f_l : V \to [0,1]^2$, $l \in [k\Delta]$ as follows. Consider an $l \in [k\Delta]$.

1. For $i \in [m]$, we set
$$f_l(u_i) = \left(\alpha_i, 2 - \frac{1}{k} - \alpha_i\right)$$
2. For $i \in [m]$ and $v_{i,j} \in V_i$, we set $f_l(v_{i,j}) = (0,0)$ if $v_{i,j} \neq v_{i,l}$. We set
$$f_l(v_{i,l}) = \left(1 - \alpha_i, \alpha_i + \frac{1}{k} - 1\right)$$

3. For $v \in V_0$, we set $f_l(v) = (0,0)$.

We verify that these embeddings satisfy the conditions (A) and (C). Fix an $l \in [k\Delta]$.

(A) Consider a set $S \in \mathcal{S}$. Let $i \in [m]$ be such that $\{u_i\} = S \cap U$. We have

$$\begin{aligned} f_l(S) &= \sum_{v \in S} f_l(v) \\ &\leq \sum_{v \in \{u_i\} \cup V_i} f(v) \\ &= f_l(u_i) + f_l(v_{i,l}) \\ &= \left(\alpha_i, 2 - \frac{1}{k} - \alpha_i\right) + \left(1 - \alpha_i, \alpha_i + \frac{1}{k} - 1\right) = (1,1). \end{aligned}$$

(C) This follows directly from the fact that $\|f_l(v)\|_1 \leq \frac{1}{k}$ for all $l \in [k\Delta]$ and $v \in V \setminus U$.

Let $g : V \to [0,1]^{2k\Delta}$ be defined as $g = (f_1, \ldots, f_{k\Delta})$. As each of the individual embeddings satisfies (A) and (C), g also satisfies the conditions (A) and (C).

Let $S \subseteq V \setminus V_0$, $|S \cap U| = 1$ be such that

$$\|g(S)\|_\infty \leq 1.$$

i.e. $\|f_l(S)\|_\infty \leq 1$ for all $l \in [k\Delta]$. Suppose that $S \cap U = \{u_i\}$. Then, we claim that $S \subseteq \{u_i\} \cup V_i$. Suppose for contradiction that this is not the case, and there exists $v_{i',l} \in V_{i'}$ with $i' \neq i, i' \in [m]$ and $l \in [k\Delta]$ such that $v_{i',l} \in S$. We have

$$\begin{aligned} f_l(S) &= \sum_{v \in S} f_l(v) \\ &\geq f_l(u_i) + f_l(v_{i',l}) \\ &= \left(\alpha_i, 2 - \frac{1}{k} - \alpha_i\right) + \left(1 - \alpha_{i'}, \alpha_{i'} + \frac{1}{k} - 1\right) \\ &= (1 + \alpha_i - \alpha_{i'}, 1 + \alpha_{i'} - \alpha_i) \end{aligned}$$

As $\alpha_i \neq \alpha_{i'}$, $\|f_l(S)\|_\infty > 1$, a contradiction. Thus, for every set $S \subseteq V$ such that $u_i \in S$, $S \cap V_{i'} \neq \phi$ for some $i' \neq i$, we have $\|g(S)\|_\infty > 1$.

Step 2. Pinning down the intra-sunflower sets. In the second step, our goal is to find an embedding $g' : V \to [0,1]^{K_2}$ with $K_2 = (k\Delta)^2$ such that

1. g' satisfies the conditions (A) and (C).
2. For every $i \in [m]$ and "intra-sunflower" set $S \subseteq \{u_i\} \cup V_i$ such that $u_i \in S$ and $S \notin \mathcal{S}^{\downarrow}$, we have $\|g'(S)\|_\infty > 1$.

We achieve this by setting $g' = (g_1, g_2, \ldots, g_{(k\Delta)^2)})$ where each g_l, $l \in [(k\Delta)^2]$ satisfies the conditions (A) and (C), and the overall function g' satisfies the second condition above.

For every $i \in [m]$, we order all the pairs of distinct elements $x, y \in V_i$ as $\{V_{i,1}, V_{i,2}, \ldots, V_{i,(k\Delta)^2}\}$ (with repetitions if needed). The upper bound on the number of such pairs is obtained using the fact that $|V_i| \leq k\Delta$ for all $i \in [m]$.

We define the embeddings $g_l : V \to [0,1]$, $l \in [(k\Delta)^2]$ below. Fix an $l \in [(k\Delta)^2]$.

1. Consider an $i \in [m]$. We have two different cases:
   (a) If $V_{i,l} \cup \{u_i\} \in \mathcal{S}^{\downarrow}$, we set $g_l(u_i) = 0$ and $g_l(v) = 0$ for all $v \in V_i$.
   (b) If $V_{i,l} \cup \{u_i\} \notin \mathcal{S}^{\downarrow}$, we set $g_l(v) = \frac{1}{k}$ for all $v \in V_{i,l}$, and $g_l(v) = 0$ for all $v \in V_i \setminus V_{i,l}$. We set
   $$g_l(u_i) = 1 - \frac{2}{k} + \frac{1}{k^2}$$
2. For all $v \in V_0$, we set $g_l(v) = 0$.

We now verify that these embeddings satisfy the conditions (A) and (C). Fix an integer $l \in [(k\Delta)^2]$.

(A) Consider a set $S \in \mathcal{S}$. Let $\{u_i\} = S \cap U$. If $\{u_i\} \cup V_{i,l} \in \mathcal{S}^{\downarrow}$, $g_l(v) = 0$ for all $v \in S$, and thus we have $|g_l(S)| \leq 1$. Now suppose that $\{u_i\} \cup V_{i,l} \notin \mathcal{S}^{\downarrow}$. This implies that $V_{i,l}$ is not a subset of S. As $|V_{i,l}| = 2$, $|V_{i,l} \cap S| \leq 1$. We get

$$\begin{aligned}
\sum_{v \in S} g_l(v) &= g_l(u_i) + \sum_{v \in S \cap V_i} g_l(v) \\
&= g_l(u_i) + \sum_{v \in S \cap V_{i,l}} g_l(v) \\
&\leq g_l(u_i) + \frac{1}{k} \\
&= 1 - \frac{2}{k} + \frac{1}{k^2} + \frac{1}{k} \leq 1
\end{aligned}$$

(C) This follows from the fact that $g_l(v) \leq \frac{1}{k}$ for all $v \in V \setminus U$.

Suppose that a set $S \subseteq V$ satisfies $S \subseteq \{u_i\} \cup V_i$ for some $i \in [m]$, and $u_i \in S$, $S \notin \mathcal{S}^{\downarrow}$. Then, we claim that $\|g'(S)\|_\infty > 1$. Suppose for the sake of contradiction that $\|g'(S)\|_\infty \leq 1$. Then, we have $g_l(S) \leq 1$ for all $l \in [(k\Delta)^2]$. Let $S = \{u_i, s_1, s_2, \ldots, s_p\}$ where $s_j \in V_i$ for all $j \in [p]$. Note that for every $v \in V_i$, there is exactly one set $S(v) \in \mathcal{S}$ such that $v \in S(v)$ and this set $S(v)$ satisfies $u_i \in S(v)$. This follows from the definition of V_i and the fact that the set family $\mathcal{S}$ is a sunflower-bouquet.

We now claim that $S(s_{j_1}) = S(s_{j_2})$ for all $j_1, j_2 \in [p]$. Suppose for contradiction that there exist $j_1, j_2 \in [p]$ with $S(s_{j_1}) \neq S(s_{j_2})$. This implies that $\{u_i, s_{j_1}, s_{j_2}\} \notin \mathcal{S}^{\downarrow}$ as otherwise, if there exists $T \in \mathcal{S}$ such that $\{u_i, s_{j_1}, s_{j_2}\} \subseteq T$, we have $S(s_{j_1}) = S(s_{j_2}) = T$. Let $l \in [(k\Delta)^2]$ be such that $V_{i,l} = \{s_{j_1}, s_{j_2}\}$. As $V_{i,l} \cup \{u_i\} \notin \mathcal{S}^{\downarrow}$, we have $g_l(v) = \frac{1}{k}$ for all $v \in V_{i,l}$ and

$$g_l(u_i) = 1 - \frac{2}{k} + \frac{1}{k^2}$$

Thus, we get that

$$\begin{aligned}\sum_{v\in S} g_l(v) &= g_l(u_i) + \sum_{v\in S\setminus\{u_i\}} g_l(v)\\ &= g_l(u_i) + \sum_{v\in V_{i,l}} g_l(v)\\ &= 1 - \frac{2}{k} + \frac{1}{k^2} + \frac{2}{k} = 1 + \frac{1}{k^2}\end{aligned}$$

contradicting the fact that $g_l(S) \leq 1$. This completes the proof that $S(s_{j_1}) = S(s_{j_2})$ for all $j_1, j_2 \in [p]$. Thus, there exists a set $S(s_1) \in \mathcal{S}$ such that $S \subseteq S(s_1)$, which implies that $S \in \mathcal{S}^{\downarrow}$, a contradiction. Thus, for every set $S \subseteq V$ such that $u_i \in S$, $S \subseteq \{u_i\} \cup V_i$ for some $i \in [m]$ and $\|g'(S)\|_\infty \leq 1$, we have $S \in \mathcal{S}^{\downarrow}$.

Final embedding. We define the final embedding $f : V \to [0,1]^{2+2k\Delta+(k\Delta)^2}$ as $f = (f_0, g, g')$. As each of these embeddings satisfies the conditions (A) and (C), the final embedding f also satisfies the conditions (A) and (C).

Suppose that $\|f(S)\|_\infty \leq 1$ for a set $S \subseteq V$. Then, $\|f_0(S)\|_\infty \leq 1$, $\|g(S)\|_\infty \leq 1$ and $\|g'(S)\|_\infty \leq 1$. Condition (D) follows immediately as $\|f_0(S)\|_\infty \leq 1$ implies that $|S| \leq k$.

We now return to condition (B). Suppose that $S \subseteq V$ with $S \cap U \neq \emptyset$ satisfies $\|f(S)\|_\infty \leq 1$. Our goal is to show that $S \in \mathcal{S}^{\downarrow}$. We have already deduced from $\|f_0(S)\|_\infty \leq 1$ that $|S \cap U| \leq 1$. As $S \cap U \neq \phi$, we have $|S \cap U| = 1$, and by using $\|f_0(S)\|_\infty \leq 1$ again, we get that $S \cap V_0 = \phi$. Let $S \cap U = \{u_i\}$. As $\|g(S)\|_\infty \leq 1$, using the argument in the first step, we can conclude that $S \cap V_{i'} = \phi$ for all $i' \neq i$. Thus, $S \subseteq \{u_i\} \cup V_i$. By using the argument in the second step, $\|g'(S)\|_\infty \leq 1$ implies that $S \in \mathcal{S}^{\downarrow}$.

Note that our construction is explicit, and we have a polynomial time algorithm to output the required embedding. The dimension of the embedding is $2 + 2k\Delta + (k\Delta)^2$, which is at most $(k\Delta)^{O(1)}$. □

As a corollary, we bound the packing dimension of the set family

$$\mathcal{T}^{\downarrow} = \mathcal{S}^{\downarrow} \cup \{S \subseteq V \setminus U : |S| \leq k\}.$$

Corollary 136. *Suppose that $\mathcal{T}$ is a set family defined on a universe V with*

$$\mathcal{T} = \mathcal{S} \cup \{S \subseteq V \setminus U : |S| \leq k\}$$

where $\mathcal{S} \subseteq 2^V$ is a sunflower-bouquet with core U. Furthermore, each set in $\mathcal{S}$ has cardinality at most $k \geq 2$ and each element appears in at most Δ sets in $\mathcal{S}$. Then,

$$\mathsf{pdim}(\mathcal{T}^{\downarrow}) \leq (k\Delta)^{O(1)}$$

Furthermore, an embedding realizing this packing dimension can be found in time polynomial in $|V|$ given $\mathcal{S}$.

Proof. As $\mathcal{S}$ is a sunflower-bouquet, from Lemma 135, there exists an embedding $f : V \to [0,1]^K$ that satisfies the conditions $(A), (B), (C)$ and (D) with $K = (k\Delta)^{O(1)}$. Conditions (A) and (C) together imply that

$$\|f(S)\|_\infty \leq 1$$

for all $S \in \mathcal{T}$. Note that

$$\mathcal{T}^\downarrow = \mathcal{S}^\downarrow \cup \{S \subseteq V \setminus U : |S| \leq k\}.$$

Suppose that $S \subseteq V$ is a subset of V with $S \notin \mathcal{T}^\downarrow$. If $S \cap U = \phi$, then $|S| > k$, which implies that $\|f(S)\|_\infty > 1$ using condition (D). If $S \cap U \neq \phi$, then $S \notin \mathcal{S}^\downarrow$ which implies that $\|f(S)\|_\infty > 1$ using condition (B). Thus, $\|f(S)\|_\infty \leq 1$ if and only if $S \in \mathcal{T}^\downarrow$. $\square$

We are now ready to prove our main embedding result i.e. Theorem 133.

Proof of Theorem 133. We define a graph $G = (V, E)$ as follows: two elements $u, v \in V$ are adjacent in G if there exist sets $S_1, S_2 \in \mathcal{S}$ (not necessarily distinct) such that $u \in S_1, v \in S_2, S_1 \cap S_2 \neq \emptyset$. As the cardinality of each set in $\mathcal{S}$ is at most k and each element of V is present in at most Δ sets, the maximum degree of a vertex in G can be bounded above as

$$\Delta(G) \leq k(k-1)\Delta^2$$

Thus, the chromatic number of G is at most $L = \chi(G) \leq k(k-1)\Delta^2 + 1 \leq k^2\Delta^2$. Using the greedy coloring algorithm, we can partition V into L non-empty parts $U_1, U_2, \ldots, U_L$ such that each U_j is a independent set in G. For every $j \in [L]$, as U_j is an independent set in G, we have

1. For every set $S \in \mathcal{S}$, $|S \cap U_j| \leq 1$.
2. Any two sets $S_1, S_2 \in \mathcal{S}$ with $S_1 \cap U_j \neq \emptyset$, $S_2 \cap U_j \neq \emptyset$ and $S_1 \cap S_2 \neq \emptyset$ satisfy $S_1 \cap U_j = S_2 \cap U_j = S_1 \cap S_2$.

We now define the set families $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_L$ as follows:

$$\mathcal{S}_j = \{S \in \mathcal{S} : S \cap U_j \neq \emptyset\} \cup \{S \subseteq V \setminus U_j : |S| \leq k\}$$

We claim that $\bigcap_{j \in [L]} \mathcal{S}_j^\downarrow = \mathcal{S}^\downarrow$. First, consider an arbitrary set $S \in \mathcal{S}^\downarrow$ and an integer $j \in [L]$. As $|S| \leq k$, irrespective of S intersects U_j or not, $S \in \mathcal{S}_j^\downarrow$. Thus, $\mathcal{S}^\downarrow \subseteq \mathcal{S}_j^\downarrow$ for all $j \in [L]$. Consider a non-empty set $S \notin \mathcal{S}^\downarrow$. As $U_1, U_2, \ldots, U_L$ is a partition of V, there exists a $j \in [L]$ such that $S \cap U_j \neq \emptyset$. As $S \notin \mathcal{S}^\downarrow$, $S \notin \mathcal{S}_j^\downarrow$. This implies that

$$\bigcap_{j \in [L]} \mathcal{S}_j^\downarrow = \mathcal{S}^\downarrow$$

Using Proposition 132, in order to bound the packing dimension of $\mathcal{S}^\downarrow$, it suffices to bound the packing dimension of $\mathcal{S}_j^\downarrow$, $j \in [L]$.

Fix an integer $j \in [L]$ and consider the set family $\mathcal{S}_j^\downarrow$. It is defined on the universe V and there exists a non-empty subset $U_j \subseteq V$ such that

$$\mathcal{S}_j = \mathcal{S}'_j \cup \{S \subseteq V \setminus U_j : |S| \leq k\}$$

with

$$\mathcal{S}'_j = \{S \in \mathcal{S} : S \cap U_j \neq \emptyset\}.$$

Here, $\mathcal{S}'_j$ is a simple set system which satisfies the following properties:

1. Each set in $\mathcal{S}'_j$ has cardinality at most $k \geq 2$ and each element appears in at most Δ sets in $\mathcal{S}'_j$.
2. Every set $S \in \mathcal{S}'_j$ satisfies $|S \cap U_j| = 1$. As $\mathcal{S}$ is non-trivial, for every $u \in U_j$, there exists a set $S \in \mathcal{S}'_j$ with $u \in S$.
3. For every pair of sets $S_1, S_2 \in \mathcal{S}'_j$ with $S_1 \cap S_2 \neq \phi$, $S_1 \cap U_j = S_2 \cap U_j = S_1 \cap S_2$.

In other words, the set family $\mathcal{S}'_j$ is a sunflower-bouquet with core U_j. Using Corollary 136, we get that $\mathsf{pdim}(\mathcal{S}_j^{\downarrow}) \leq (k\Delta)^{O(1)}$ for all $j \in [L]$, which completes the proof. □

8.3.3 Hardness of Vector Bin Packing

We show that for large enough constant d, Vector Bin Packing is hard to approximate within $\Omega(\log d)$. Our hardness is obtained via the hardness of set cover on simple bounded instances.

In the set cover problem, the input is a set family $\mathcal{S}$ on a universe V with $|V| = n$. The objective is to pick the minimum number of sets $\{S_1, S_2, \ldots, S_m\} \subseteq \mathcal{S}$ from the family such that their union is equal to V. The greedy algorithm where we repeatedly pick the set that covers the maximum number of new elements achieves a $\ln n$ approximation factor. Fiege [Fei98] proved a matching hardness of $(1 - \epsilon)(\ln n)$. On set systems where each pair of sets intersect in at most one element i.e. simple instances, $\Omega(\log n)$ hardness of set cover is proved by Kumar, Arya, and Ramesh [KAR00]. We observe that by changing the parameters slightly, their reduction also implies the same hardness on instances where the maximum set size is bounded:

Theorem 137. *(Set Cover on simple bounded instances) There exists an integer B_0 such that for every constant $B \geq B_0$, the Set Cover problem on simple set systems in which each set has cardinality at most B is NP-hard to approximate within $\Omega(\log B)$. Furthermore, in the hard instances, each element occurs in at most $O(B)$ sets.*

The details of the parameter modification appear in Section 8.5.

We combine this set cover hardness with the bound on the packing dimension of simple set systems to prove the hardness of Vector Bin Packing.

Proof of Theorem 120. We prove the result by giving an approximation preserving reduction from the NP-hard problem of set cover on simple bounded set systems. Let $\mathcal{S}$ be the set system from Theorem 137 defined on a universe V. Note that each set in the family has cardinality at most $k = B$ and each element in the universe appears in at most $\Delta = O(B)$ sets. We now output a set $\mathcal{V}$ of $|V|$ vectors in $[0, 1]^d$ such that

1. (Completeness.) If there is a set cover of size m in $\mathcal{S}$, there is a packing of $\mathcal{V}$ using m bins.
2. (Soundness.) If there is no set cover of size m' in $\mathcal{S}$, there is no packing of $\mathcal{V}$ using m' bins.

We use Theorem 133 to compute an embedding $f : V \to [0,1]^d$ in polynomial time such that

$$\|f(S)\|_\infty \leq 1$$

if and only if $S \in \mathcal{S}^\downarrow$, with $d = (k\Delta)^{O(1)} = B^{O(1)}$. Our output Vector Bin Packing instance is the set of vectors $f(v), v \in V$.

$$\mathcal{V} = \{f(v) : v \in V\}$$

Completeness. Suppose that there exist sets $S_1, S_2, \ldots, S_m \in \mathcal{S}$ whose union is V. Then, we use m bins with the vectors $\{f(v_j) : j \in S_i\}$ in the ith bin. A vector might appear in multiple bins, but we can arbitrarily pick one bin for each vector while still maintaining the property that in each bin, the ℓ_∞ norm of the sum of the vectors is at most 1.

Soundness. Suppose that the minimum set cover in $\mathcal{S}$ has cardinality at least $m' + 1$. Then, we claim that the set of vectors $\mathcal{V}$ needs $m' + 1$ bins to be packed. Suppose for contradiction that there is a vector packing with m' bins. In other words, there exists a partition of V into $B_1, B_2, \ldots, B_{m'}$ such that $\|f(B_i)\|_\infty \leq 1$ for all $i \in [m']$. As $\|f(B_i)\|_\infty \leq 1$, $B_i \in \mathcal{S}^\downarrow$ for all $i \in [m']$. That is, for every $i \in [m']$, there exists a set $S_i \in \mathcal{S}$ such that $B_i \subseteq S_i$. This implies that $\{S_1, S_2, \ldots, S_{m'}\}$ is a set cover of V, a contradiction.

As the original bounded simple set cover problem is hard to approximate within $\Omega(\log B) = \Omega(\log d)$, the resulting Vector Bin Packing is hard to approximate within $\Omega(\log d)$. Furthermore, in the hard instances, the optimal value i.e. the minimum number of bins needed to pack the vectors can be made arbitrarily large, and thus, the hardness applies to the asymptotic approximation ratio. □

8.4 Vector Scheduling

8.4.1 Monochromatic Clique

In the Monochromatic Clique problem, given a graph $G = ([n], E)$ and a parameter $k(n)$, the objective is to assign k colors to the vertices of G so as to minimize the largest monochromatic clique. More formally, we study the following decision version of the problem.

Definition 138. *(*Monochromatic-Clique(k, B)*) In the* Monochromatic-Clique(k, B) *problem, given a graph $G = (V, E)$ with $|V| = n$ and parameters $k(n), B(n)$, the goal is to distinguish between the following:*

1. *(YES case) The chromatic number of G is at most k.*
2. *(NO case) In any assignment of k colors to the vertices of G, there is a clique of size B, all of whose vertices are assigned the same color.*

It generalizes the standard k-Coloring problem, which corresponds to the case when $B = 2$. Note that the problem gets easier as B increases. Indeed, when $B > \sqrt{n}$, we can solve the problem in polynomial time using the canonical SDP relaxation. We present this algorithm and an almost matching integrality gap in Section 8.6.

On the hardness front, we now prove that $\textsf{Monochromatic-Clique}(k, B)$ is hard when $B = (\log n)^C$, for any constant C. We achieve this in two steps: First, we observe that the existing chromatic number hardness results already imply the hardness of monochromatic clique when $B = (\log n)^\gamma$ for some constant $\gamma > 0$. Next, we amplify this hardness by using lexicographic graph product.

Basic Hardness

We start with a couple of basic Ramsey theoretic lemmas from [CK04].

Lemma 139. *For a graph $G = (V, E)$ with $|V| = n$, if $\omega(G) \leq B$, then $\alpha(G) \geq n^{\frac{1}{B}}$.*

Lemma 140. *For a graph $G = (V, E)$ with $|V| = n$, if $\omega(G) \leq B$, then $\chi(G) \leq O(n^{1-\frac{1}{B}} \log n)$.*

We can use the above lemmas to prove that if the chromatic number of a graph is large enough, then in any assignment of k colors to the vertices of the graph, there is a large monochromatic clique.

Lemma 141. *For every constant $\epsilon > 0$, if a graph $G = (V, E)$ with $|V| = n$ satisfies $\chi(G) \geq k\frac{n}{2^{(\log n)^\alpha}}$ for some integer k and $0 < \alpha < 1$, then in any assignment of k colors to V, there is a monochromatic clique of size $B = \Omega\left((\log n)^{1-\alpha-\epsilon}\right)$.*

Proof. Suppose for contradiction that there is an assignment of k colors V without a monochromatic clique of size B. Using Lemma 140, the subgraphs corresponding to each of the k color classes has chromatic number at most

$$O(n^{1-\frac{1}{B}} \log n) = \frac{n}{2^{\Omega((\log n)^{\alpha+\epsilon})}} \log n < \frac{n}{2^{(\log n)^\alpha}}$$

colors. Thus, the whole graph has chromatic number at most $k\frac{n}{2^{(\log n)^\alpha}}$ colors, a contradiction. $\square$

Khot [Kho01] proved that assuming $\mathsf{NP} \not\subseteq \mathsf{ZPTIME}\left(n^{(\log n)^{O(1)}}\right)$, the chromatic number of graphs is hard to approximate within a factor of $\frac{n}{2^{(\log n)^{1-\gamma}}}$ for an absolute constant $\gamma > 0$. More formally, he proved the following:

Theorem 142. *([Kho01]) There exists a constant $\gamma > 0$, a function $k = k(n)$, and a randomized reduction that takes as input a 3-SAT instance I on n variables and outputs a graph $G = (V, E)$ with $|V| = N = 2^{\log n^{O(1)}}$ such that*

1. *(Completeness) If I is satisfiable, $\chi(G) \leq k$.*
2. *(Soundness) If I is not satisfiable, with probability at least $\frac{1}{2}$, $\chi(G) > k\frac{N}{2^{(\log N)^{1-\gamma}}}$.*

Futhermore, the reduction runs in time $\mathsf{poly}(N) = 2^{(\log n)^{O(1)}}$.

We observe that Khot's chromatic number hardness immediately gives $(\log n)^{\Omega(1)}$ hardness of Monochromatic Clique.

Lemma 143. *There exists a constant $\gamma > 0$, a function $k = k(n)$ such that the following holds. Assuming $\mathsf{NP} \not\subseteq \mathsf{ZPTIME}\left(n^{(\log n)^{O(1)}}\right)$, given a graph $G = ([n], E)$, there is no $n^{(\log n)^{O(1)}}$ time algorithm for $\textsf{Monochromatic-Clique}(k, B)$ when $B = \Omega\left((\log n)^\gamma\right)$.*

Proof. Using Khot's reduction, we get that there exists an absolute constant $\gamma > 0$ such that assuming $\mathsf{NP} \not\subseteq \mathsf{ZPTIME}\left(n^{(\log n)^{O(1)}}\right)$, given a graph $G = ([n], E)$ and a parameter $k(n)$, there is no $n^{(\log n)^{O(1)}}$ time algorithm to distinguish between the following:

1. (Completeness) $\chi(G) \leq k$.
2. (Soundness) $\chi(G) > k \frac{n}{2^{(\log n)^{1-\gamma}}}$.

Using Lemma 141, the Soundness condition implies that in any assignment of k colors to G, there is a monochromatic clique of size $\Omega\left((\log n)^{\gamma-\epsilon}\right)$, for any constant $\epsilon > 0$. Thus, given a graph G and a parameter k, assuming $\mathsf{NP} \not\subseteq \mathsf{ZPTIME}\left(n^{(\log n)^{O(1)}}\right)$, there is no $n^{(\log n)^{O(1)}}$ time algorithm to distinguish between the following:

1. (Completeness) $\chi(G) \leq k$.
2. (Soundness) In any assignment of k colors to the vertices of G, there is a monochromatic clique of size $\Omega((\log n)^{\gamma'})$.

for any constant $\gamma' < \gamma$. □

Amplification using Lexicographic Product

We cannot directly amplify the hardness of the Monochromatic-Clique problem by taking graph products as we cannot preserve the chromatic number and also amplify the largest clique in an assignment of k colors at the same time. We get around this issue by defining a harder variant of Monochromatic Clique called Strong Monochromatic Clique and then amplifying it.

Definition 144. *(*Strong Monochromatic-Clique(k, B, C)*) In the* Strong Monochromatic-Clique(k, B, C)*, given a graph G and parameters $k(n), B(n), C$, the goal is to distinguish between the following two cases:*

1. *(YES case) The chromatic number of G is at most k.*
2. *(NO case) In any assignment of k^C colors to the vertices of G, there is a monochromatic clique of size B.*

We now observe that the chromatic number hardness of Khot [Kho01] implies the same hardness as Lemma 143 for Strong Monochromatic Clique as well.

Lemma 145. *There exists a constant $\gamma > 0$ and a function $k = k(n)$ such that for every constant $C \geq 1$, the following holds. Assuming $\mathsf{NP} \not\subseteq \mathsf{ZPTIME}\left(n^{(\log n)^{O(1)}}\right)$, there is no $n^{(\log n)^{O(1)}}$ time algorithm for* Strong Monochromatic-Clique(k, B, C) *when $B = \Omega((\log n)^{\gamma})$.*

Proof. Note that the function k in Theorem 142 satisfies $k = o\left(2^{(\log N)^{1-\gamma}}\right)$. Thus, we can replace the soundness condition in Theorem 142 with $\chi(G) \geq k^C \frac{N}{2^{C(\log N)^{1-\gamma}}}$. Using Lemma 141, this implies that in any assignment of k^C colors to the vertices of G, there is a monochromatic clique of size $\Omega((\log N)^{\gamma-\epsilon})$, where $\epsilon > 0$ is an absolute constant. The hardness of Strong Monochromatic Clique then follows along the same lines as Lemma 143. □

We amplify the hardness of Strong Monochromatic-Clique(k, B, C) to Monochromatic-Clique(k^C, B^C) using the lexicographic product of graphs. First, we define lexicographic product and prove some properties of it.

Definition 146. *(Lexicographic product of graphs) Given two graphs G and H, the Lexicographic graph product $G \cdot H$ has vertex set $V(G) \times V(H)$, and two vertices $(u_1, v_1), (u_2, v_2)$ are adjacent if either $(u_1, u_2) \in E(G)$ or $u_1 = u_2$ and $(v_1, v_2) \in E(H)$.*

The lexicographic product can be visualized as replacing each vertex of G with a copy of H and forming complete bipartite graphs between copies of vertices adjacent in G. For ease of notation, we let $G^2 = G \cdot G$. More generally, for an integer n that is a power of 2, we define G^n as taking the above lexicographic product of G with itself recursively $\log n$ times.

Lemma 147. *Let $n \geq 2$ be a power of 2. If $\chi(G) \leq k$, then $\chi(G^n) \leq k^n$.*

Proof. We prove that $\chi(G^2) \leq k^2$, and the statement follows by induction on n. If $f : G \to [k]$ is a proper k-coloring of G, then the coloring $f'(u, v) = (f(u), f(v))$ is a proper k^2-coloring of $G \times G$. □

Lemma 148. *Let $n \geq 2$ be a power of 2. Suppose that in any assignment of k colors to the vertices of G, there is a monochromatic clique of size B. Then, in any assignment of k colors to the vertices of G^n, there is a monochromatic clique of size B^n.*

Proof. We prove the statement for $n = 2$ and the lemma follows by induction on n. Let $f : V(G^2) \to [k]$ be a given assignment. For a vertex $v \in G$, consider the assignment $g_v : V(G) \to [k]$ defined as $g_v(u) = f(v, u)$. As every assignment of k colors to the vertices of G has a monochromatic clique of size B, there is a color $\alpha(v) \in [k]$ and a clique $S(v) \subseteq V(G)$ with $|S(v)| \geq B$ such that $g_v(u) = \alpha(v)$ for all $u \in S(v)$, or in other words, $f(v, u) = \alpha(v)$ for all $u \in S(v)$. Note that such a set $S(v)$ and $\alpha(v)$ exist for $v \in V(G)$. The function $\alpha : V(G) \to [k]$ can also be visualized as an assignment of k colors to the vertices of G, and thus there is a monochromatic clique T of size at least B with respect to this assignment. The set

$$\{S(v) : v \in T\}$$

is a monochromatic clique of size B^2 with respect to f in G. □

By using the lexicographic product, we can get a polynomial time reduction from Strong Monochromatic Clique to Monochromatic Clique.

Lemma 149. *For every constant $C \geq 1$ that is a power of 2, there exists a polynomial time reduction from* Strong Monochromatic-Clique(k, B, C) *to* Monochromatic-Clique(k^C, B^C).

Proof. Given a graph G as an instance of Strong Monochromatic-Clique(k, B, C), we compute the graph $G' = G^C$. We claim that solving Monochromatic-Clique(k^C, B^C) on G' solves the original Strong Monochromatic Clique problem.

1. (Completeness.) Suppose that $\chi(G) \leq k$. Then, by Lemma 147, $\chi(G') \leq k^C$.

2. (Soundness.) Suppose that in any assignment of k^C colors to the vertices of G, there is a monochromatic clique of size B. Then, by Lemma 148, in any assignment of k^C colors to the vertices of G', there is a monochromatic clique of size B^C. $\square$

Putting everything together, we obtain the following hardness of Monochromatic Clique.

Theorem 150. *For every constant $C > 0$, there exists a function $k = k(n)$ such that the following holds. Assuming* NP $\not\subseteq$ ZPTIME $\left(n^{(\log n)^{O(1)}}\right)$*, there is no $n^{(\log n)^{O(1)}}$ time algorithm for* Monochromatic-Clique(k, B) *when $B = \Omega\left((\log n)^C\right)$.*

Proof. The proof follows directly by combining Lemma 145 and Lemma 149. $\square$

8.4.2 From Monochromatic Clique to Vector Scheduling

We now prove Theorem 121 using the above hardness of Monochromatic Clique.

Proof of Theorem 121. The reduction from Monochromatic-Clique(k, B) to Vector Scheduling is (implicitly) proved in [CK04]. We present it here for the sake of completeness. Given a graph $G = (V = [n], E)$, parameters k and B, we order all the B-sized cliques of G as $T_1, T_2, \ldots, T_d$ with $d \leq n^B$. We define a set of n vectors $v_1, v_2, \ldots, v_n$ of dimension d with

$$(v_i)_j = \begin{cases} 1 & \text{if } i \in T_j \\ 0 & \text{otherwise.} \end{cases}$$

The instance of the Vector Scheduling has these n vectors as the input and the number of machines is equal to k.

We analyze the reduction.

1. (Completeness.) Suppose that there exists a proper k-coloring of G, $c : V \to [k]$. We assign the vector v_i to the machine $c(i)$. For every $j \in [d]$, all the B vectors that have 1 in the jth dimension are assigned to distinct machines. Thus, the makespan of the scheduling is at most 1.
2. (Soundness.) Suppose that in any assignment of k colors to the vertices of G, there is a monochromatic clique of size B. In this case, the makespan of the scheduling is at least B.

We set $B = (\log n)^C$ for a large constant C to be set later. We choose k from Theorem 150 such that assuming NP $\not\subseteq$ ZPTIME $\left(n^{(\log n)^{O(1)}}\right)$, there is no $n^{(\log n)^{O(1)}}$ time algorithm for Monochromatic-Clique(k, B). By the above reduction, we can conclude that there is no polynomial time algorithm that approximates the resulting Vector Scheduling instances within a factor of $B = (\log n)^C$. As $d \leq n^B$, we get that $\log d \leq (\log n)^{C+1}$, and $B \geq (\log d)^{1-\frac{1}{C+1}}$. Setting $C = \frac{1}{\epsilon} - 1$, we get that d-dimensional Vector Scheduling has no polynomial time $\Omega((\log d)^{1-\epsilon})$ approximation algorithm assuming NP $\not\subseteq$ ZPTIME $\left(n^{(\log n)^{O(1)}}\right)$, for every constant $\epsilon > 0$. $\square$

Remark 151. *In [Im+19], Im, Kell, Kulkarni, and Panigrahi also study the ℓ_r-norm minimization of Vector Scheduling where the objective is to minimize*

$$\max_{k\in[d]}\left(\sum_{i=1}^{m}(L_i^k)^r\right)^{\frac{1}{r}}$$

where L_i^k denotes the load on the machine i on the kth dimension. They gave an algorithm with an approximation ratio $O\left(\left(\frac{\log d}{\log\log d}\right)^{1-\frac{1}{r}}\right)$. Our reduction from Monochromatic Clique gives almost optimal hardness for this variant as well: we get the hardness of $\Omega\left((\log d)^{1-\frac{1}{r}-\epsilon}\right)$ assuming NP $\not\subseteq$ ZPTIME $\left(n^{(\log n)^{O(1)}}\right)$, for every constant $\epsilon > 0$.

8.4.3 Hardness of Vector Scheduling via Balanced Hypergraph Coloring

Observe that the resulting Vector Scheduling instances in the above reduction satisfy a stronger property: the vectors are from $\{0,1\}^d$. In the setting where the vectors are from $\{0,1\}^d$, the Vector Scheduling problem is closely related to the Balanced Hypergraph Coloring problem. In this problem, given a hypergraph H and an integer k, the objective is to assign k colors to the vertices of H minimizing the maximum number of monochromatic vertices in an edge. More formally, we study the following decision version of the problem.

Definition 152. *(Balanced Hypergraph Coloring.) In the Balanced Hypergraph Coloring problem, given a s-uniform hypergraph H and parameters k and $c < s$, the objective is to distinguish between the following:*

1. *There is an assignment of k colors to the vertices of H such that in every edge, each color appears at most c times.*
2. *The hypergraph H has no proper coloring with k colors i.e., in any assignment of k colors to the vertices of H, there is an edge all of whose s vertices are assigned the same color.*

We give a simple reduction from Balanced Hypergraph Coloring to Vector Scheduling.

Lemma 153. *Given a s-uniform hypergraph $H = (V' = [n'], E')$ and parameters k, c, there is a polynomial time reduction that outputs a Vector Scheduling instance I over n' vectors $v_1, v_2, \ldots, v_{n'} \in \{0,1\}^d$ on m' machines with $m' = k, d = |E'|$ such that*

1. *(Completeness.) If there is an assignment of k colors to the vertices of H such that each color appears at most c times in every edge, then there is a scheduling of I with makespan at most c.*
2. *(Soundness.) If H has no proper coloring with k colors, then in any scheduling of I, the makespan is at least s.*

Proof. Let $d = |E'|$. Order the edges of the hypergraph H as $e_1, e_2, \ldots, e_d$. We define the set of vectors $v_1, v_2, \ldots, v_{n'} \in \{0,1\}^d$ as follows:

$$(v_i)_j = \begin{cases} 1 & \text{if } i \in e_j \\ 0 & \text{otherwise.} \end{cases}$$

We set the number of machines m' to be equal to the number of colors k. There is a natural correspondence between the assignment of k-colors to the vertices of H $f : V' \to [k]$, and the scheduling where we assign the vector v_i to the machine $f(i)$. We now analyze our reduction.

1. (Completeness.) If there exists an assignment of k colors $f : V' \to [k]$ where each color appears at most c times in each edge, we assign the vector $v_i, i \in [n']$ to the machine $f(i)$. In any dimension $j \in [d]$, at most c vectors v_i with $(v_i)_j = 1$ are scheduled on any machine. Thus, in any machine, the total load in each dimension is at most c.
2. (Soundness.) If there exists a vector scheduling $f : [n] \to [m']$ with makespan strictly smaller than s, assign the color $f(i)$ to the ith vertex of the hypergraph. In any edge of the hypergraph, each color appears fewer than s times as the makespan is smaller than s. Thus, $f : V' \to [k]$ is a proper k-coloring of the hypergraph H. □

We prove the hardness results for Vector Scheduling, namely Theorem 122 and Theorem 123 by combining this reduction with the hardness of Balanced Hypergraph Coloring. Note that the dimension of the resulting instances in the above reduction is equal to m, the number of edges in the hypergraph H, and the ratio of the makespans in the completeness and soundness is equal to $\frac{s}{c}$. Thus, our goal is to prove the hardness of the Balanced Hypergraph Coloring problem where $\frac{s}{c}$ is as large as possible, as a function of m, the number of edges in the underlying hypergraph.

Towards this, we first give a reduction from the Label Cover problem to the Balanced Hypergraph Coloring problem.

Lemma 154. *Fix an odd prime number $k \geq 3$ and let $\epsilon = \frac{1}{k^8}$. Given a Label Cover instance $G = (V = L \cup R, E, \Sigma_L, \Sigma_R, \Pi)$, there is a polynomial time reduction that outputs a k^2 uniform hypergraph $H = (V', E')$ with $|V'| \leq |L|k^{|\Sigma_L|}$ such that*

1. *(Completeness) If G is satisfiable, there is an assignment of k colors to the vertices of H such that in every edge, each color occurs at most $2k$ times.*
2. *(Soundness) If no labeling to G can satisfy an ϵ fraction of the constraints, then H has no proper k-coloring, that is, in any assignment of k colors to the vertices of H, there is an edge all of whose vertices are assigned the same color.*

Furthermore, $|E'|$ is at most $|R|\Delta^k k^{|\Sigma_L|k^2}$ where Δ is the maximum degree of a vertex $v \in R$.

We defer the proof of Lemma 154 to Section 8.4.4.

Using Lemma 154, we can prove the hardness of Balanced Hypergraph Coloring via Label Cover hardness results. We obtain two different hardness results for the Balanced Hypergraph Coloring problem, one under $\mathsf{NP} \nsubseteq \mathsf{DTIME}\left(n^{O(\log\log n)}\right)$ and another NP-hardness result, by using two different hardness results for the Label Cover problem. These two hardness results prove Theorem 123 and Theorem 122 respectively, using Lemma 153.

First, using the standard Label Cover hardness obtained using PCP Theorem [Aro+98] combined with Raz's Parallel Repetition theorem [Raz98], we get the following hardness of Balanced Hypergraph Coloring.

Theorem 155. *Assuming $\mathsf{NP} \nsubseteq \mathsf{DTIME}\left(n^{O(\log\log n)}\right)$, there is no polynomial time algorithm for the following problem. Given a k^2-uniform hypergraph $H = (V', E')$ with $m = |E'|$ and $k = (\log m)^{\Omega(1)}$, distinguish between the following:*

1. *There is an assignment of k colors to the vertices of H such that in any edge of the hypergraph, each color appears at most $2k$ times.*
2. *The hypergraph H has no proper k coloring.*

Proof. By setting $\epsilon = \frac{1}{k^8}$ in Theorem 129, we have a reduction from the 3-SAT problem on n variables to the Label Cover problem $G = (V = L \cup R, E, \Sigma_L, \Sigma_R, \Pi)$ with soundness ϵ and $|V| \leq n^{O(\log k)}$, $|\Sigma_L| \leq k^{O(1)}$ and $\Delta \leq k^{O(1)}$. Using Lemma 154, we can reduce this Label Cover instance to a Balanced Hypergraph Coloring instance $H = (V', E')$ with $|V'| \leq n^{O(\log k)} 2^{k^{O(1)}}$ and $|E'| \leq n^{O(\log k)} 2^{k^{O(1)}}$. We set $k = (\log n)^{\Omega(1)}$ such that $|V'| = n^{O(\log \log n)}$ and $|E'| = n^{O(\log \log n)}$ to obtain the required hardness of Balanced Hypergraph Coloring. □

The proof of Theorem 123 follows immediately from Theorem 155 and Lemma 153.

Next, using the hardness of near linear sized Label Cover due to Moshkovitz and Raz [MR10], we obtain the following NP-hardness of Balanced Hypergraph Coloring.

Theorem 156. *For any constant $C \geq 1$, given a k^2 uniform hypergraph $H = (V', E')$ with $m = |E'|$ and $k = (\log \log m)^C$, it is NP-hard to distinguish between the following:*

1. *There is an assignment of k colors to the vertices of H such that in any edge of the hypergraph, each color appears at most $2k$ times.*
2. *The hypergraph H has no proper k coloring.*

Proof. By setting $\epsilon = \frac{1}{k^8}$ in Theorem 130, we can reduce a 3-SAT instance over n variables to a Label Cover instance $G = (V = L \cup R, E, \Sigma_L, \Sigma_R, \Pi)$ with soundness ϵ and $|V| \leq n^{1+o(1)} k^{O(1)}$, $|\Sigma_L| \leq 2^{k^{O(1)}}$, $\Delta = k^{O(1)}$. By using Lemma 154, we can reduce the Label Cover instance to a Balanced Hypergraph Coloring instance $H = (V', E')$ with $|V'| \leq n^{1+o(1)} 2^{2^{k^{O(1)}}}$ and $|E'|$ at most $n^{1+o(1)} 2^{2^{k^{O(1)}}}$. We set $k = (\log \log n)^{\Omega(1)}$ to obtain $|V'| = O(n^2), |E'| = O(n^2)$.

Dinur and Steurer [DS14] gave an improvement to [MR10]–in the new Label Cover hardness, the alphabet size $|\Sigma_L|$ can be taken to be $2^{\left(\frac{1}{\epsilon}\right)^\gamma}$ for *every* constant $\gamma > 0$. Using this improved Label Cover hardness, we can set $k = (\log \log n)^C$ for any constant $C \geq 1$ in the hardness of Balanced Hypergraph Coloring. □

The proof of Theorem 122 follows immediately from Theorem 156 and Lemma 153.

Finally, we remark that if the structured graph version of the Projection Games Conjecture [Mos15] holds, Lemma 154 and Lemma 153 together prove that d-dimensional Vector Scheduling is NP-hard to approximate within a factor of $(\log d)^{\Omega(1)}$.

8.4.4 Proof of Lemma 154

We follow the standard Label Cover-Long Code framework–see e.g.,[ABP20].

Reduction. For ease of notation, let $n = |\Sigma_L|$. For every node $v \in L$ of the Label Cover instance, we have a set of k^n vertices denoted by $f_v = \{v\} \times [k]^n$. The vertex set of the hypergraph is $V' = \bigcup_{v \in L} f_v$.

For every $u \in R$, and k distinct neighbors of u, $v_1, v_2, \ldots, v_k \in L$ with projection constraints $\pi_i : [\Sigma_L] \to [\Sigma_R], i \in [k]$, consider the set of k^2 vectors $\mathbf{x}^{i,j}$ for $i \in [k], j \in [k]$ which satisfy the following: For every $\beta \in \Sigma_R$, and for all $\alpha_1, \alpha_2, \ldots, \alpha_k \in \Sigma_L$ such that $\pi_i(\alpha_i) = \beta$ for all $i \in [k]$, we have

$$\left|\{(i,j) | \mathbf{x}^{i,j}_{\alpha_i} = p\}\right| \leq 2k \ \forall p \in [k] \tag{8.1}$$

For every such set of k^2 vectors, we add the edge $\{(v_i, \mathbf{x}^{i,j}) : 1 \leq i, j \leq k\}$ to E'. We can observe that $|V'| \leq |L| k^{|\Sigma_L|}$ and

$$|E'| \leq |R| \binom{\Delta}{k} \binom{k^{|\Sigma_L|}}{k}^k \leq |R| \Delta^k k^{|\Sigma_L| k^2}.$$

Completeness. Suppose that there exists an assignment $\sigma : V \to \Sigma$ that satisfies all the constraints of the Label Cover instance G. We color the set of vertices f_v in the long code corresponding to the vertex $v \in L$ with the dictator function on the coordinate $\sigma(v)$ i.e. for every $\mathbf{x} \in f_v$, we assign the color

$$c\left(\{v, \mathbf{x}\}\right) = \mathbf{x}_{\sigma(v)}$$

We can observe that this coloring satisfies the property that in every edge $e \in E'$, each color appears at most $2k$ times.

Soundness. Suppose that there is a proper k-coloring $c : V' \to [k]$ of the hypergraph H i.e. in every edge $e = \{v_1, v_2, \ldots, v_{k^2}\}$, we have

$$|\{c(v_1), c(v_2), \ldots, c(v_{k^2})\}| > 1$$

Our goal is to prove that there is a labeling to the Label Cover instance that satisfies at least $\epsilon = \frac{1}{k^8}$ fraction of constraints.

We need the following lemma proved by Austrin, Bhangale, Potukuchi [ABP20] using a generalization of Borsuk-Ulam theorem.

Lemma 157. *(Theorem 5.2 of [ABP20]) For every odd prime k and $n \geq k^3$, in any k-coloring of $[k]^n$, $c : [k]^n \to [k]$, there is a set of k vectors $\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^k$ that are all assigned the same color such that*

$$\{\boldsymbol{x}^1_i, \boldsymbol{x}^2_i, \ldots, \boldsymbol{x}^k_i\} = [k]$$

for at least $n - k^3$ distinct coordinates $i \in [n]$.

Using this lemma, for every $v \in L$, we can identify a set of vectors $\mathbf{x}^{v,1}, \mathbf{x}^{v,2}, \ldots, \mathbf{x}^{v,k} \in f_v$ such that all these vectors have the same color i.e. $c(\{v, \mathbf{x}^{v,i}\}) = c'(v)$ for all $v \in L, i \in [k]$ for some function $c' : L \to [k]$. Furthermore, there are a set of coordinates $S(v) \subseteq [n]$ with $|S(v)| \leq k^3$ such that

$$\{\mathbf{x}^{v,1}_i, \mathbf{x}^{v,2}_i, \ldots, \mathbf{x}^{v,k}_i\} = [k]$$

for every $i \in [n] \setminus S(v)$.

For a set $S \subseteq \Sigma_L$ and a function $\pi : \Sigma_L \to \Sigma_R$, we use $\pi(S)$ to denote the set $\{\pi(i) : i \in S\}$. We now prove a key lemma that helps in the decoding procedure.

Lemma 158. *Let $u \in R$ be a node on the right side of the Label Cover instance. There are a set of labels $S(u) \subseteq \Sigma_R$ such that $|S(u)| \leq k^5$, and for every $v \in L$ that is a neighbor of u with projection constraint $\pi : \Sigma_L \to \Sigma_R$, we have $S(u) \cap \pi(S(v)) \neq \phi$.*

Proof. Fix a node $u \in R$ on the right side of the Label Cover instance. Let $v_1, v_2, \ldots, v_l \in L$ be the neighbors of u in the Label Cover instance corresponding to the projection constraints $\pi_1, \pi_2, \ldots, \pi_l$ respectively. As $|S(v_i)| \leq k^3$ for all $i \in [l]$, and the constraints π_i are projections, we have $|\pi_i(S(v_i))| \leq k^3$ for all $i \in [l]$. Among these l subsets $\pi_i(S(v_i))$ of Σ_R, let the maximum number of pairwise disjoint subsets be denoted by l'. Without loss of generality, we can assume that $\mathcal{S} = \{\pi_i(S(v_i)) : i \in [l']\}$ is a pairwise disjoint family of subsets.

We define the set $S(u)$ as follows:

$$S(u) = \bigcup_{i \in [l']} \pi_i(S(v_i))$$

As $\mathcal{S}$ is a family of maximum pairwise disjoint subsets, we have $S(u) \cap \pi_i(S(v_i)) \neq \phi$ for all $i \in [l]$. Our goal is to bound the size of $S(u)$, which we achieve by bounding l'.

We claim that $l' \leq k(k-1)$. Suppose for contradiction that $l' > k(k-1)$. This implies that there are $l' > k(k-1)$ nodes $v_1, v_2, \ldots, v_{l'}$ all adjacent to u such that $\pi_i(S(v_i)), i \in [l']$ are all pairwise disjoint. Thus, there exists a color $\ell \in [k]$ and a set of k nodes $w_1, w_2, \ldots, w_k$ adjacent to u corresponding to the projection constraints $\pi'_1, \pi'_2, \ldots, \pi'_k$ such that $c'(w_i) = \ell$ for all $i \in [k]$, and the k sets $\pi'_i(S(w_i))$ are pairwise disjoint.

Using this, we can construct a set of vectors $\mathbf{x}^{i,j}, 1 \leq i, j \leq k$ defined as $\mathbf{x}^{i,j} = \mathbf{x}^{w_i,j}$ which satisfy the following properties:

1. All these vectors are colored the same:
$$c(\{w_i, \mathbf{x}^{i,j}\}) = \ell \ \forall 1 \leq i, j \leq k$$
2. For every $i \in [k]$,
$$\{\mathbf{x}^{i,1}_{i'}, \mathbf{x}^{i,2}_{i'}, \ldots, \mathbf{x}^{i,k}_{i'}\} = [k]$$
for every $i' \in [n] \setminus S(w_i)$.

We claim that these set of vectors satisfy the condition in Equation (8.1). Fix a $\beta \in \Sigma_R$, and $\alpha_1, \alpha_2, \ldots, \alpha_k \in \Sigma_L$ such that $\pi'_i(\alpha_i) = \beta$ for all $i \in [k]$. As the family of subsets $\pi'_i(S(w_i))$ is a pairwise disjoint family, we can infer that there exists at most one $i \in [k]$ such that $\alpha_i \in S(w_i)$. Note that if $\alpha_i \notin S(w_i)$, then
$$\{\mathbf{x}^{i,j}_{\alpha_i} : j \in [k]\} = [k].$$
Thus, we have
$$\left|\{(i,j) | \mathbf{x}^{i,j}_{\alpha_i} = p\}\right| \leq 2k \ \forall p \in [k].$$
Thus, the set of vectors $\{(w_i, \mathbf{x}^{i,j}) : 1 \leq i, j \leq k\}$ is indeed an edge of E'. As all these vectors are colored the same color ℓ, we have arrived at a contradiction to the fact that c is a proper k-coloring of H.

Hence, we can conclude that $l' \leq k(k-1)$, and thus, $|S(u)| \leq k(k-1)k^3 < k^5$. □

Now, consider the labeling $\sigma : L \to \Sigma_L$, where $\sigma(v), v \in L$ is chosen uniformly at random from $S(v)$. Similarly, let $\sigma : R \to \Sigma_R$ is chosen uniformly at random from $S(u), u \in R$. Using Lemma 158, we can infer that for every edge $e = (v, u)$ in the Label Cover, this labeling satisfies the edge e with probability at least $\frac{1}{|S(v)||S(u)|} \geq \frac{1}{k^8}$. By linearity of expectation, this labeling satisfies at least $\frac{1}{k^8}$ fraction of the constraints in expectation. Hence, with positive probability, the labeling satisfies at least $\frac{1}{k^8}$ fraction of the constraints. This concludes the proof of soundness that if H has a proper k coloring, then there exists a labeling to G that satisfies at least $\frac{1}{k^8}$ fraction of the constraints.

8.5 Hardness of simple k-set cover

The hardness result of Kumar, Arya, and Ramesh [KAR00] is obtained from the Label Cover problem using a partition gadget along the lines of the reduction of Lund and Yannakakis [LY94]. The set families in the reduction in [LY94] have large intersections. [KAR00] get around this by using two main ideas:

1. They use a different partition system wherein each partition is a disjoint union of a large (super constant) number of sets instead of just 2 sets in [LY94].
2. They use multiple sets for each label assignment to a vertex of the Label Cover, unlike a single set corresponding to each label of each vertex in [LY94].

As [KAR00] were proving a $\Omega(\log n)$ hardness of the set cover, the universe size of the partition system is chosen to be the same as the number of vertices in the Label Cover instance. This forces the set sizes to be very large. We can get around this issue by simply defining the partition system on a set of size B, where B is a large constant. This also has an added benefit that we no longer require sub-constant hardness from the Label Cover instances, thus giving us NP-hardness directly. This observation is used by Trevisan [Tre01] to obtain $\ln B - O(\ln \ln B)$ NP-hardness of set cover on instances where each set has cardinality at most B, from Feige's $(1 - \epsilon) \ln n$ set cover hardness [Fei98].

We now describe the parameter modifications in full detail. Let B be a large constant.

We start our reduction from Label Cover instances with soundness $\gamma = \frac{1}{32\beta^2 \log^2 B}$ where β is an absolute constant to be fixed later.

Theorem 159. *([Aro+98; Raz98]) Given a Label Cover instance defined on a bipartite graph $G = (V, E)$ with left alphabet Σ_L and right alphabet Σ_R, it is NP-hard to distinguish between the following:*

1. *(Completeness). There exists a labeling $\sigma : V \to \Sigma_L \cup \Sigma_R$ that satisfies all the constraints.*
2. *(Soundness). No labeling to V can satisfy more than γ fraction of the constraints.*

Furthermore the instances satisfy the following properties:

1. *The alphabet sizes $d = |\Sigma_L|$ and $d' = |\Sigma_R|$ are both upper bounded by $(\log B)^{O(1)}$.*
2. *The maximum degree deg of G is upper bounded by $(\log B)^{O(1)}$.*

Following the convention in [KAR00], we assume that the number of vertices on the left side in G is equal to that on the right side of G, and we denote this number by n'.

We now construct a partition system $\mathcal{P}$ on a universe N of size B. The system $\mathcal{P}$ has $d' \times (deg+1) \times d$ partitions. Each partition has $m = B^{1-\epsilon}$ parts, where ϵ is a small constant to be fixed later. The partition system is divided into d' groups each containing $(deg+1) \times d$ partitions. Each group is further organized into $deg + 1$ subgroups each of which contains d partitions. Let $P_{g,s,p}$ denote the pth partition in the sth subgroup of the gth group and $P_{g,s,p,k}$ denote the kth set in $P_{g,s,p}$ where $g \in [d'], s \in [deg+1], p \in [d], k \in [m]$. The partition system satisfies the four properties in Section 4 of [KAR00], the only difference being that the universe N now has size B instead of n'. Thus, the covering property (Property 4 in [KAR00]) now states that any covering of N with $\beta m \log B$ sets should contain at least $\frac{3m}{4}$ sets from the same partition. Such a partition system is shown to exist for large enough B in [KAR00] using a randomized construction. They also derandomize the construction. But for our setting, as B is a constant, we just need to show the existence of such a partition system.

We reduce the Label Cover instance in Theorem 159 to a set cover instance $\mathcal{SC}$ by the same construction as in [KAR00]: we have a partition system corresponding to each edge of the Label Cover instance, and the union of the elements in the partition systems is the element set of $\mathcal{SC}$. The sets in $\mathcal{SC}$, $C_k(v, a)$ are defined exactly as in [KAR00]. The cardinality of each set is at most $B' = deg \times B \leq B^2$. Each element is present in at most $md = O(B)$ sets. The fact that $\mathcal{SC}$ is a simple set system follows from Lemma 1 of [KAR00]. By Lemma 2 in [KAR00], if there is a labeling of the Label Cover instance, then there is a set cover of size $n'm$ in $\mathcal{SC}$. If there is a set cover of size $\frac{\beta}{2} n'm \log B$ in $\mathcal{SC}$, then there is a labeling of G that satisfies γ fraction of constraints. The proof of this soundness follows along the same lines as Lemma 3 of [KAR00], with the only difference being that we now define the *good* edges as edges having $\#(e) \leq \beta m \log B$.

8.6 SDP Relaxation of Monochromatic-Clique

We consider the following SDP relaxation of the graph coloring problem on $G = (V, E)$:

$$\begin{aligned}
&\text{Minimize } k \\
&\quad \langle u_i, u_i \rangle = 1 \ \forall i \in V \\
&\quad \langle u_i, u_j \rangle \leq \frac{-1}{k-1} \ \forall (i,j) \in E
\end{aligned}$$

The optimal solution to this SDP is referred to as the vector chromatic number $\chi_v(G)$ of the graph G. It is equivalent to the Lovasz theta function of the complement of G. We have the following sandwich property due to [Knu94]:

$$\omega(G) \leq \chi_v(G) \leq \chi(G)$$

8.6.1 Algorithm when $B > \sqrt{n}$

There is a simple algorithm for the Monochromatic-Clique(k, B) problem when $B > k$: We compute $\chi_v(G)$ in polynomial time, and we check if $\chi_v(G) \leq k$. In this case, there is no clique of size B in G, and we output YES. If $\chi_v(G) > k$, then the graph cannot be colored with k colors, and in this case, we output NO.

Note that if $k(B-1) \geq n$, there is always an assignment of k colors to the vertices of the graph without a clique of size B, thus the problem is trivial.

8.6.2 Integrality gap

The above algorithm proves that in any graph with vector chromatic number at most k, there is an assignment of k colors to the vertices that has monochromatic clique of size at most $\sqrt{n}$. We now prove that this cannot be significantly improved:

Theorem 160. *For n large enough, there exists a graph $G = (V, E)$ with n vertices, and a parameter k such that*

1. $\chi_v(G) \leq k$.
2. *In any assignment of k colors to the vertices of G, there is a monochromatic clique of size $n^{\Omega(1)}$.*

Proof. We first prove the following: for large enough n, there exists a graph G on n vertices, and an integer k such that

1. $\vartheta(G) \leq k$.
2. In any assignment of k colors to the vertices of the graph G, there exists a monochromatic independent set of size $B = n^{\Omega(1)}$.

Our construction is a probabilistic one: we sample G from $G(n, p)$ with $p = \frac{1}{\sqrt{n}}$. It has been proved [Juh82] that the Lovasz theta function of this random graph satisfies

$$\vartheta(G) \leq 2n^{\frac{3}{4}} + \tilde{O}(n^{\frac{1}{3}} \log n)$$

with high probability. We set $k = 3n^{\frac{3}{4}}$. For large enough n, with high probability, we have $\vartheta(G) \leq k$.

Furthermore, the random graph $G(n, p)$ with $p = o(n^{-\frac{2}{5}})$ has no K_6 with high probability(See e.g., [FK15]). Thus, using Lemma 139, we can infer that in any subset of size $\frac{n^{\frac{1}{4}}}{3}$, there is an independent set of size at least $\frac{n^{\frac{1}{24}}}{2}$. Hence, in any assignment of k colors to the vertices of the graph G, there is a monochromatic independent set of size $n^{\Omega(1)}$. Taking the complement, we get a graph with the required properties. $\square$

Chapter 9

Approximate hypergraph vertex cover and generalized Tuza's conjecture

9.1 Introduction

The relationship between minimum vertex covers and maximum matchings of graphs and hypergraphs is a fundamental and well-studied topic in combinatorics and optimization. Even though the worst-case factor t gap between the two parameters cannot be improved on arbitrary t-uniform hypergraphs, there are some interesting special cases where the ratio between these quantities is smaller. A classic example of this phenomenon is the König's theorem on bipartite graphs, where the sizes of minimum vertex covers and maximum matchings are equal.

For the case of $t = 3$, a notorious open problem capturing this gap on special 3-uniform hypergraphs is Tuza's conjecture[Tuz81; Tuz90], which states that in any graph, the number of edges required to hit all triangles is at most twice the maximum number of edge-disjoint triangles. For a hypergraph H, let us denote by $\tau(H)$ and $\nu(H)$ the sizes of the minimum vertex cover and maximum matching respectively. Tuza's conjecture is then equivalent to the statement $\tau(H) \leq 2 \cdot \nu(H)$ for any 3-uniform hypergraph H obtained by taking the edges of a graph G as its vertices, and the triangles of G as its (hyper)-edges. (Taking $G = K_4$ shows that the factor 2 is best possible.) The conjecture has been verified for various classes of graphs such as graphs without $K_{3,3}$-subdivision [Kri95], graphs with maximum average degree less than 7 [Pul15], graphs with quadratic number of edge disjoint triangles [HR01; Yus12], graphs with treewidth at most 6[BFG19], and random graphs in the $G_{n,p}$ model [BCD20; KP20]. On general graphs, the current best upper bound on the ratio is a factor of 2.87 due to Haxell [Hax99].

Aharoni and Zerbib [AZ20] introduced an extension of Tuza's conjecture to hypergraphs of larger uniformity. This generalized Tuza's conjecture states that for any t-uniform hypergraph H, the minimum vertex cover $\tau(H')$ of $H' = H^{(t-1)}$ is at most $\lceil \frac{t+1}{2} \rceil$ times that of the maximum matching $\nu(H')$. Here, for a t-uniform hypergraph $H = (V, E)$, the $(t-1)$-*blown-up hypergraph* $H' = H^{(t-1)}$ is a t-uniform hypergraph whose vertices are the set of all $(t-1)$ sized subsets that are contained in at least one edge of H, and corresponding to every edge e in H, all the $(t-1)$-sized subsets of e form an edge in H'. Tuza's conjecture is a special case of their conjecture when $t = 3$ and H has hyperedges corresponding to the triangles in a graph. As is the case with the original Tuza's conjecture, the conjectured value of $\left\lceil \frac{t+1}{2} \right\rceil$ is the best possible gap: when H is the complete t-uniform hypergraph on $(t+1)$ vertices, the $(t-1)$-blown-up hypergraph $H' = H^{(t-1)}$ has $\nu(H') = 1$ and $\tau(H') = \left\lceil \frac{t+1}{2} \right\rceil$.

9.1.1 Fractional Tuza's conjecture and the algorithmic hypergraph Turán problem

A first step towards non-trivially bounding $\tau(H)$ in terms of $\nu(H)$ for hypergraphs H from some structured family of hypergraphs is proving its *fractional version*, i.e., obtaining the same upper bound on the ratio between $\tau(H)$ and $\nu^*(H)$, the fractional maximum matching size. By LP duality, this is equivalent to bounding the ratio between $\tau(H)$ and $\tau^*(H)$, the fractional vertex cover value. As $\nu(H) \leq \nu^*(H) = \tau^*(H) \leq \tau(H)$ for any hypergraph H, establishing the fractional version is a necessary step toward bounding $\tau(H)/\nu(H)$. Note that understanding the extremal ratio between τ and τ^* on a given family of hypergraphs is equivalent to bounding the integrality gap of the natural linear programming relaxation of vertex cover on that class of hypergraphs.

Krivelevich[Kri95] proved the fractional version of Tuza's conjecture that $\tau(H^{(2)}) \leq 2\tau^*(H^{(2)})$ for any 3-uniform hypergraph H. A multi-transversal version of Krivelevich's result is proved in a recent work [Cha+20]. In this work, we prove the fractional version of the generalized Tuza's conjecture (upto $o(t)$ factors), establishing a non-trivial upper bound on the LP integrality gap for $(t-1)$-blown-up hypergraphs.

Theorem 161. *For any t-uniform hypergraph H, $\tau(H') \leq \left(\frac{t}{2} + 2\sqrt{t \ln t}\right) \tau^*(H')$, where $\tau(H')$ and $\tau^*(H')$ are respectively the size of the minimum vertex cover and minimum fractional vertex cover of the blown-up hypergraph $H' = H^{(t-1)}$. Furthermore, there is an efficient algorithm to approximate vertex cover on $(t-1)$-blown-up hypergraphs within a $\frac{t}{2} + 2\sqrt{t \ln t}$ factor.*

The vertex cover problem on $(t-1)$-blown-up hypergraphs is also intimately connected to the famous Hypergraph Turán Problem [Tur41; Tur61] in extremal combinatorics. In the Hypergraph Turán Problem, the goal is to find the minimum size of a family $\mathcal{F} \subseteq \binom{[n]}{t-1}$ of subsets of $[n]$ with cardinality $(t-1)$ such that for every subset S of $[n]$ of size t, there exists a set $T \in \mathcal{F}$ such that T is a subset of S. The best known upper bound is due to [Sid97]: there exists a family $\mathcal{F} \subseteq \binom{[n]}{t-1}$ of size $O(\frac{\log t}{t})\binom{n}{t-1}$ such that for every subset S of $[n]$ of size t, there exists a subset $T \in \mathcal{F}$ such that T is contained in S. On the other hand, the lower bound situation is rather dire, with only second-order improvements [CL99; LZ09] over the trivial $\frac{1}{t}\binom{n}{t-1}$ lower bound.

Note that the Hypergraph Turán Problem is precisely the minimum vertex cover problem on $H^{(t-1)}$ when H is the *complete* t-uniform hypergraph. Thus, for a general hypergraph H, finding vertex covers on the blown-up hypergraph $H^{(t-1)}$ can be viewed as an algorithmic version of the Hypergraph Turán problem.

Problem 162. *(Algorithmic Hypergraph Turán Problem (AHTP)) Given a t-uniform hypergraph $H = (V = [n], E)$, find the minimum size of a family $\mathcal{F} \subseteq \binom{[n]}{t-1}$ of subsets of V of size $(t-1)$ such that for every hyperedge $e \in E$, there exists $T \in \mathcal{F}$ such that T is a subset of e.*

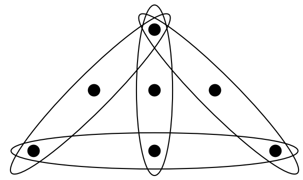

Figure 9.1: The 3-tent

The problem is a generalization of the minimum vertex cover on graphs, which corresponds to the case $t = 2$. As AHTP can be cast as a vertex cover problem on t-uniform hypergraphs, there is a trivial factor t approximation algorithm for this problem. We prove Theorem 161 by obtaining an improved algorithm for AHTP based on rounding the standard LP relaxation on $H' = H^{(t-1)}$.

We now briefly describe this rounding approach. First, using threshold rounding, we argue that one may focus on the case when the LP solution does not have any variables that are assigned values greater than $\frac{2}{t}$. Let S be the set of vertices of H' that are assigned non-zero LP value. The thresholding procedure ensures that every hyperedge $e \in E(H')$ intersects with S in at least $\frac{t}{2}$ vertices. We can bound the cardinality of S from above by $t \cdot \mathsf{OPT}$ using the dual matching LP, where OPT is the cost of the optimal LP solution. Our goal then becomes finding a vertex cover of size at most about $\frac{|S|}{2}$. We achieve this by a color-coding technique: we randomly assign a color from $\{0, 1\}$ to each vertex of H independently. Most edges of H are almost balanced under this coloring, in the sense that each color appears at least $t/2 - o(t)$ times. We then use this balance property to find a small vertex cover in H'.

9.1.2 Vertex cover vs. matching and excluded sub-hypergraphs

The generalized Tuza's conjecture concerns the relationship between τ and ν on the $(t-1)$-blown-up hypergraphs. There have been some works in the literature on the gap between τ and ν on other structured class of hypergraphs. An outstanding result of this type is Aharoni's proof [Aha01] that $\tau(H) \leq 2\nu(H)$ for all tripartite 3-uniform hypergraphs H. Aharoni and Zerbib [AZ20] asked if there is a structural explanation that unites the generalized Tuza's conjecture and the above result—for example, does the exclusion of a certain substructure in the hypergraph H imply better gaps between $\tau(H)$ and $\nu(H)$. A particular substructure they studied is the "tent" subhypergraph (Figure 9.1).

They observed that both tripartite hypergraphs and 2-blown-up 3-uniform hypergraphs cannot contain the tent as a subhypergraph, and asked whether a generalization of Tuza's conjecture might hold for 3-uniform hypergraphs that exclude tents. If this is the case, it could give a common structural explanation of the existence of small vertex covers in 3-uniform hypergraphs.

In this work, we answer this question in the negative. We prove that that there are hypergraphs H on n vertices that exclude tents with $\tau(H) \geq (1 - o(1))n$. Since $\nu(H) \leq n/3$ trivially, this shows that the ratio τ/ν can approach 3 on tent-free 3-uniform hypergraphs, and the extension of Tuza's conjecture as raised in [AZ20] does not hold. More generally, one might ask if there is some collection of hypergraphs which are excluded from blown-up hypergraphs whose absence implies a non-trivial gap between τ and ν. In fact, we prove a stronger statement showing that there is no 3-uniform hypergraph family $\mathcal{F}$ (that is absent from blown-up hypergraphs) whose exclusion alone could imply Tuza's conjecture. Our result applies for larger uniformity t and the fractional version of Tuza's conjecture.

Theorem 163. *For every $\epsilon > 0$ and every finite family of t-uniform hypergraphs $\mathcal{F}$ such that no hypergraph from $\mathcal{F}$ appears in any $(t-1)$-blown-up hypergraph $H' = H^{(t-1)}$, there is a t-uniform hypergraph T such that T does not contain any hypergraph from $\mathcal{F}$ but $\tau(T) \geq (t-\epsilon)\nu(T)$ (and in fact $\tau(T) \geq (t-\epsilon)\tau^*(T)$).*

The above result rules out the possibility of a "local" proof of Tuza's conjecture. Our construction is a probabilistic one, first sampling each edge of the hypergraph independently with certain probability, and then removing all the copies of hypergraphs in $\mathcal{F}$. Using the fact that the family $\mathcal{F}$ satisfies certain sparsity requirements [FM08; BFM10], we can conclude that there is no large independent set in this construction.

We also provide an *explicit construction* that answers the tent-free question of [AZ20]: our counterexample is the hypergraph T with vertex set $[3]^n$ for large enough n and edges being the set of combinatorial lines. By the density Hales Jewett Theorem [FK91; Pol12], there is no large independent set in T, and using the structure of combinatorial lines, we can prove that T does not have any tent.

9.1.3 Vertex cover and set cover on simple hypergraphs

As mentioned earlier, AHTP is a special case of vertex cover on t-uniform hypergraphs. In fact, the blown-up hypergraph $H^{(t-1)}$ is a *simple*[1] hypergraph: any two edges intersect in at most one vertex. This is simply because any two distinct t-sized subsets of $[n]$ intersect in at most one $(t-1)$-sized subset. Simple hypergraphs have been well studied in Graph Theory, especially in the context of Erdős-Faber-Lovász conjecture [Erd81; Erd88] which has been recently proved in a breakthrough result [Kan+21], Ryser's conjecture [Fra+17] and chromatic number of bounded degree hypergraphs [DLR95; FM13].

A natural question is whether we can obtain an approximation ratio smaller than t for vertex cover on simple hypergraphs. However,Theorem 163 shows that the natural LP has an integrality gap approaching t on simple, and indeed a lot more structured, hypergraphs. But perhaps there are other algorithms that beat the trivial factor t approximation for this problem. We prove that this is not the case, and in fact, vertex cover on simple hypergraphs is as hard as vertex cover on general t-uniform hypergraphs.

Theorem 164. *For every $\epsilon > 0$, unless $\mathsf{NP} \subseteq \mathsf{BPP}$, no polynomial time algorithm can approximate vertex cover on simple t-uniform hypergraphs within a factor of $t-1-\epsilon$. Under the Unique Games conjecture, the inapproximability factor improves to $t-\epsilon$.*

We also study the set cover problem on simple set families where any two sets in the family intersect in at most one element. Equivalently, we want to pick the minimum number of edges to cover all vertices in a simple hypergraph. Kumar, Arya, and Ramesh [KAR00] proved that

[1] Simple hypergraphs are also referred to as linear hypergraphs.

set cover problem on simple set systems is hard to approximate within a $\Omega(\ln n)$ factor. While this is within constant factor of the $\ln n$ approximability of set cover on general set systems, it is natural to wonder if the set cover problem on simple systems is as hard as the same problem on general set systems. Contrary to the vertex cover problem, it turns out that simplicity of the set family does help in getting an improved approximation factor for the set cover—in fact, the greedy algorithm itself delivers such an approximation.

Theorem 165. *For set cover on simple set systems over a universe of size n, the greedy algorithm achieves an approximation ratio $\frac{\ln n}{2} + 1$. Further, there are simple set systems where the greedy algorithm is off by a factor exceeding $\frac{\ln n}{2} - 1$.*

Interestingly, the dual Maximum Coverage problem, where the goal is to cover as many elements as possible with a specified number of sets, does not become easier on simple set systems and is hard to approximate within a factor exceeding $(1 - 1/e)$ [CKL21], the factor achieved by the greedy algorithm on general set systems. In [CKL20], the authors conjecture the hardness of achieving an approximation factor beating $(1 - 1/e)$ even for the Maximum Coverage version of AHTP, and call this the *Johnson Coverage Hypothesis*. They show that this hypothesis implies strong inapproximability results for fundamental clustering problems like k-means and k-median on Euclidean metrics. For example, they showed that the hypothesis implies that k-median is hard to approximate within a factor of 1.73 on ℓ_1 metrics, matching the best hardness factor on general metrics due to Guha and Khuller [GK99].

9.1.4 Other improved hypergraph vertex cover algorithms

Algorithms beating the trivial factor t approximation have been obtained for the vertex cover problem on some other families of t-uniform hypergraphs. In his doctoral thesis, Lovász [Lov75] gave a LP rounding algorithm to obtain a factor $\frac{t}{2}$ approximation for vertex cover on t-uniform t-partite hypergraphs. This algorithm is shown to be optimal under the Unique Games Conjecture by Guruswami, Sachdeva, and Saket [GSS15], and an almost matching NP-hardness is also shown. Aharoni, Holzman, and Krivelevich [AHK96] generalized the above algorithmic result to other class of hypergraphs which have a partition of vertices obeying certain size restrictions. A factor $\frac{t}{2}$ approximation algorithm has also been obtained on subdense regular t-uniform hypergraphs [Car+12].

For the problem of covering all paths of length t (t-Path Transversal), Lee [Lee19] gave a factor $O(\log t)$ approximation. For covering all copies of the star on t vertices, i.e., $K_{1,t-1}$, a factor $O(\log t)$ approximation is given in [GL17], and this is tight by a simple reduction from dominating set on degree t graphs. Covering 2-connected t-vertex pattern graphs (in particular t-cliques or t-cycles) is as hard as general t-uniform hypergraph vertex cover [GL17].

9.1.5 Open problems

A number of intriguing questions and directions come to light following our work, and we mention a few of them below.

The most obvious question is whether our algorithm for AHTP can be improved and yield approximation ratios smaller than $t/2$. Can stronger LP relaxations like Sherali-Adams help in this regard? On the hardness side, essentially nothing is known. There is a straightforward approximation preserving reduction from vertex cover on graphs to AHTP, but this only shows the hardness of beating a factor of 2. Can one show a better inapproximability factor? We do not know any good lower bound on the integrality gap of the LP either—for example, we do not know the existence of a hypergraph H for which $\tau(H^{(t-1)})/\tau^*(H^{(t-1)})$ grows with t. A natural candidate is the complete t-uniform hypergraph on n vertices for which de Cain conjectured [Cae94] that in fact, $\tau(H^{(t-1)})/\tau^*(H^{(t-1)})$ grows with t. However, this is precisely the lower bound of hypergraph Turán problem, and is perhaps very hard to resolve. On the algorithmic side, obtaining $o(\log t)$ approximation algorithm for AHTP would lead to improvements on the hypergraph Turán problem: On the complete hypergraph instance, either the algorithm outputs a family $\mathcal{F} \subseteq \binom{[n]}{t-1}$ of size $o\left(\frac{\log t}{t-1}\right)\binom{n}{t}$ that covers every subset of size t, or gives a certificate that any such family should have size at least $\omega\left(\frac{1}{t}\right)\binom{n}{t-1}$. In the first case, we get an improvement on the upper bound of hypergraph Turán problem, and in the second case, we resolve de Cain's conjecture.

Similar to the $(t-1)$-blown-up hypergraphs, one can define the k-blown-up hypergraph of a t-uniform hypergraph—which will be a $\binom{t}{k}$-uniform hypergraph—and study the vertex cover problem on it. A special case of this problem when $k = 2$ is the analog of Tuza's problem for larger cliques, i.e., covering all copies of t-cliques in a graph by the fewest possible edges. Our algorithm for AHTP extends to this setting, and in particular gives an algorithm with ratio $t^2/4$ for the $k = 2$ case, beating the trivial $\binom{t}{2}$ factor (see Section 9.3.4 for details). Can one achieve a $o(t^2)$ factor algorithm? A simple reduction from vertex cover on t-uniform hypergraphs shows an inapproximability factor of $t - O(1)$, but can one show hardness or integrality gaps of $\omega(t)$?

In general, our work brings to the fore challenges about covering graph structures by *edges*, on both the algorithmic and hardness fronts. On the hardness side, we seem to have essentially no techniques to show strong inapproximability results, as the known PCP techniques where one naturally associates vertices with proof locations do not seem to apply. As mentioned earlier, covering all copies of a t-vertex pattern graph H with *vertices* is as hard to approximate as general t-uniform hypergraph vertex cover when H is 2-connected [GL17].

Our structural results show that the LP integrality gap (and therefore also the vertex cover to matching ratio) remains close to t on hypergraphs that exclude subgraphs absent in $(t-1)$-blown-up hypergraphs, and thus no "local" proof of Tuza-type conjectures is possible. Are there interesting families of t-uniform hypergraphs $\mathcal{F}$ such that vertex cover admits non-trivial approximation (with ratio less than t) on $\mathcal{F}$-free t-uniform hypergraphs?

For the maximization version of AHTP, where we seek to pick a specified number $(t-1)$-sized subsets to cover the largest number of edges in a t-uniform hypergraph, is there an algorithm that beats the $(1-1/e)$ factor (achieved by greedy for the general Max Coverage problem)? The Johnson Coverage Hypothesis of [CKL20] asserts that for any $\epsilon > 0$, a $(1-1/e+\epsilon)$-approximation is hard to obtain for t large enough compared to ϵ.

We have considered covering problems in this work, and there are interesting questions concerning the dual packing problems as well. For instance, what is the approximability of packing *edge-disjoint* copies, of say t-cliques, in a graph? This is a special case of the matching problem on 2-blown-up hypergraphs. For the maximum matching problem on general k-uniform hypergraphs, also known as k-set packing, Cygan [Cyg13] gave a local search algorithm that achieves an approximation factor of $\frac{k+1}{3} + \epsilon$ for any constant $\epsilon > 0$. Can we get better algorithms for the maximum matching problem on blown-up hypergraphs?

On the hardness front, k-set packing is inapproximable to a $\Omega(k/\log k)$ factor [HSS06]. Known inapproximability results for the independent set problem on graphs with maximum degree k [AKS11; Cha16] imply that the maximum matching problem on k-uniform *simple* hypergraphs is hard to approximate within a $\Omega\left(\frac{k}{\log^2 k}\right)$ factor. Could maximum matching on simple hypergraphs be easier to approximate than general hypergraphs?

Organization. In Section 9.2, we introduce some notation and definitions. In Section 9.3, we describe and analyze our algorithm for AHTP and prove Theorem 161. Then, in Section 9.4, we prove that the analog of (generalized) Tuza's conjecture does not hold based only on local forbidden sub-hypergraph characterizations, proving Theorem 163, and also giving an explicit construction for the tent-free case posed in [AZ20]. Finally, in Section 9.5, we consider simple hypergraphs and prove Theorems 164 and 165.

9.2 Preliminaries

Notation. We use $[n]$ to denote the set $\{1, 2, \ldots, n\}$. We use $\mathbb{Z}_n$ to denote the set $\{0, 1, \ldots, n-1\}$. For a set S and an integer $1 \leq k \leq |S|$, we use $\binom{S}{k}$ to denote the family of all the k-sized subsets of S. A hypergraph $H' = (V', E')$ is called a subhypergraph of $H = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E'$. For a hypergraph $H = (V, E)$, we use $\tau(H), \nu(H)$ to denote the size of the minimum vertex cover and the maximum matching respectively. Similarly, we use $\tau^*(H)$ to denote the minimum fractional vertex cover of H:

$$\tau^*(H) = \min\left\{\sum_{v \in V} x_v : x_v \in \mathbb{R}_{\geq 0}\ \forall v \in V, \sum_{v \in e} x_v \geq 1\ \forall e \in E\right\}$$

We define the k-blown up hypergraph formally:

Definition 166. *For a t-uniform hypergraph $G = (V, E)$ and for an integer $1 \leq k < t$, we define the k-blown up hypergraph $H = G^{(k)} = (V', E')$ as follows:*

1. *The vertex set $V' \subseteq \binom{V}{k}$ is the set of all k-sized subsets of V that are contained in an edge of G:*

$$V' = \{U : U \subseteq V, |U| = k, \exists e \in E : U \subseteq e\}$$

2. *For every edge $e \in E$, we include in E' all the k-sized subsets of e, so that*

$$E' = \left\{e' : e' = \binom{e}{k}, e \in E\right\}$$

We will need the following Chernoff bound:

Lemma 167. *(Multiplicative Chernoff bound) Suppose $X_1, X_2, \ldots, X_n$ are independent random variables taking values in $\{0, 1\}$. Let $X = X_1 + X_2 + \ldots + X_n$, and let $\mu = \mathbb{E}[X]$. Then, for any* $0 \le \delta \le 1$,

$$Pr(X \le (1-\delta)\mu) \le e^{-\frac{\delta^2 \mu}{2}} \quad .$$

9.3 LP rounding algorithm for AHTP

In this section, we present our algorithm for the AHTP and prove Theorem 161. Given a t-uniform hypergraph G as an input to the AHTP, let $H = G^{(t-1)}$ be the $(t-1)$-blown-up hypergraph of G.

9.3.1 Color-coding based small vertex cover

We first prove a lemma that in any $(t-1)$-blown-up hypergraph $H = ([n], E)$, there is a vertex cover of size at most $O(\frac{\log t}{t})n$ using a color-coding argument. This lemma illustrates the color-coding idea well, and is also useful later in the context of structural characterization of the blown-up hypergraphs. This lemma is not used in the main algorithm, and the reader can skip to Section 9.3.2 for the algorithm.

Lemma 168. *Suppose $G = ([n], E(G))$ is a t-uniform hypergraph and $H = G^{(t-1)} = (V(H), E(H))$. Then, there exists a randomized polynomial time algorithm that outputs a vertex cover of H with expected size at most $|V(H)| \left(\frac{2\ln t}{t} + O\left(\frac{1}{t}\right)\right)$.*

Proof. Our algorithm is based on the color-coding technique used to get upper bounds for the hypergraph Turán problem [KR83; Sid95]. Let $P = \left\lceil \frac{t-1}{2\ln t} \right\rceil$. Color each vertex of G with $c : [n] \to [P]$ uniformly independently at random. For $v \in V(H)$ and $i \in [P]$, let $C_i(v)$ denote the number of nodes of v that are colored with i, i.e., $C_i(v) := |\{j \in v : c(j) = i\}|$.

We define a function $f : V(H) \to \mathbb{Z}_P$ as

$$f(v) = C_1(v) + 2C_2(v) + \ldots + (P-1)C_{(P-1)}(v) \mod P$$

For an element $i \in \mathbb{Z}_P$, let $f^{-1}(i)$ denote the set $\{v \in V(H) : f(v) = i\}$. Let $p \in \mathbb{Z}_P$ be such that $|f^{-1}(p)| \le |f^{-1}(i)|$ for all $i \in \mathbb{Z}_P$. Note that by definition, $|f^{-1}(p)| \le \frac{|V|}{P}$. Let $U \subseteq V(H)$ be defined as follows:

$$U = \{v : v \in V(H), \exists i \in [P] \text{ such that } C_i(v) = 0\}$$

We claim that $S = f^{-1}(p) \cup U$ is a vertex cover of H. Consider an arbitrary edge $e = \{v_1, v_2, \ldots, v_t\} \in E(H)$. Let the corresponding edge in G be equal to $e(G) = \bigcup_{j\in[t]} v_j = (u_1, u_2, \ldots, u_t) \in E(G)$ where $u_1, u_2, \ldots, u_t$ are elements of $[n]$. Without loss of generality, let $v_j = e(G) \setminus \{u_j\}$. For a color $i \in [P]$, let $C_i(e) = |\{j \in [t] : c(u_j) = i\}|$. We consider two cases separately:

1. First, if there exists a color $i \in [P]$ such that $C_i(e) = 0$, then for every $j \in [t]$, $C_i(v_j) = 0$, and thus, for every $j \in [t]$, $v_j \subseteq U$, and thus, $e \cap S \neq \phi$.

2. Suppose that for every color $i \in [P]$, $C_i(e) > 0$. We define $f(e) \in \mathbb{Z}_P$ as

$$f(e) = C_1(e) + 2C_2(e) + \ldots + (P-1)C_{(P-1)}(e) \mod P$$

Note that for every $j \in [t]$, we have

$$f(v_j) = f(e) - c(u_j) \mod P$$

As the size of $\{c(u_1), c(u_2), \ldots, c(u_t)\}$ is equal to P, the size of the set $\{f(v_1), f(v_2), \ldots, f(v_t)\}$ is equal to P as well. Thus, there exists a $j \in [t]$ such that $f(v_j) = p$ which implies that $v_j \in S$.

Thus, our goal is to upper bound the expected value of $|S|$. Note that $P \leq \frac{t-1}{\ln t}$. By taking union bound over all the colors, we get

$$\mathbb{E}[U] \leq P\left(1 - \frac{1}{P}\right)^{t-1} |V(H)| \leq \frac{t-1}{\ln t} e^{-2\ln t} |V(H)| \leq \left(\frac{1}{t \ln t}\right) |V(H)| \leq O\left(\frac{1}{t}\right) |V(H)|$$

Thus, the expected value of S is at most $|f^{-1}(p)| + \mathbb{E}[|U|]$ which is at most $\left(\frac{2\ln t}{t-1} + O\left(\frac{1}{t}\right)\right) |V(H)|$.

□

9.3.2 LP rounding based algorithm for AHTP

Consider the standard LP relaxation for vertex cover in H:

$$\text{Minimize} \sum_{v \in V(H)} x_v$$

$$\text{such that} \quad \sum_{v \in e} x_v \geq 1 \ \ \forall e \in E(H); \quad \text{and} \quad x_v \geq 0 \ \ \forall v \in V(H)$$

Let $\overline{x}$ be an optimal solution to the above Linear Program, and let $\mathsf{OPT} = \sum_{v \in V(H)} \overline{x}_v$. Let $S \subseteq V(H)$ be the set of vertices that are assigned positive LP value i.e.

$$S = \{v \in V(H) : \overline{x}_v > 0\}$$

We need a lemma relating $|S|$ and OPT:

Lemma 169. *The cardinality of S is at most $t \cdot \mathsf{OPT}$.*

Proof. Consider the dual of the vertex cover LP:

$$\text{Maximize} \sum_{e \in E(H)} y(e)$$

$$\text{such that} \sum_{e \ni v} y(e) \leq 1 \ \forall v \in V(H); \quad \text{and} \quad y(e) \geq 0 \ \forall e \in E(H)$$

Let $\overline{y}$ be an optimal solution to the above matching LP. By LP-duality, we get $\sum_{e \in E(H)} \overline{y}_e = \mathsf{OPT}$. Recall that for all $v \in S$, $\overline{x}_v \neq 0$. By the complementary slackness conditions, we get that for all $v \in S$, $\sum_{e \ni v} \overline{y}_e = 1$. Summing over all $v \in S$, we obtain

$$|S| = \sum_{v \in S} \sum_{e \ni v} \overline{y}_e \leq t \sum_{e \in E(H)} \overline{y}_e = t \cdot \mathsf{OPT}.$$

□

In general, OPT could be much smaller than $|V(H)|$, and thus we cannot use Lemma 168 directly to obtain algorithm for AHTP. However, we can obtain a simple $(t-1)$-factor approximation algorithm for AHTP using Lemma 168, extending the proof of fractional Tuza's conjecture of Krivelevich [Kri95]. We consider two different cases:

1. Suppose that there is a vertex $v \in V(H)$ such that $\overline{x}_v = 0$. Consider an arbitrary edge $e \in E(H)$ with $v \in e$. As $\sum_{u \in e} \overline{x}_u \geq 1$, we can infer that there is a vertex $v' \in e$ such that $\overline{x}_{v'} \geq \frac{1}{t-1}$. We round v' to 1 i.e. add v' to our vertex cover solution, delete all the edges containing v' and recursively proceed.
2. Suppose that for every vertex $v \in V(H)$, we have $\overline{x}_v > 0$. In this case, using Lemma 168, we can find a vertex cover of size $O(\frac{\log t}{t})|V(H)|$, which can be bounded above by $O(\log t)\mathsf{OPT}$ using Lemma 169.

We now describe a randomized algorithm to round the LP to obtain an integral solution whose expected size is at most $\left(\frac{t}{2} + 2\sqrt{t \ln t}\right) \mathsf{OPT}$. As is evident from the second case in the above $(t-1)$-factor algorithm, the problem is easy when the set of vertices $S \subseteq V(H)$ with non-zero LP value is large. Instead of considering the two different cases based on whether $S = V(H)$ or not, we take a more direct approach by finding a vertex cover of size $\left(\frac{1}{2} + o(1)\right) |S|$. Combined with Lemma 169, we get our required approximation guarantee.

For ease of notation, let $t' = \frac{t}{2} + 2\sqrt{t \ln t}$. Our first step is to round all the variables above a certain threshold to 1 (Algorithm 1). However, we need to do it recursively to ensure that we can bound the optimal value of the remaining instance.

Algorithm 1 Recursive thresholding for AHTP

1: Let $\gamma = \frac{1}{t'}$.
2: Let $\overline{x}$ be an optimal solution of the LP and let $V' = \{v : \overline{x}_v \geq \gamma\}$.
3: Let $U = V'$.
4: **while** V' is non-empty **do**
5: Delete V' from $V(H)$, and delete all the edges $e \in E(H)$ that contain at least one vertex $v \in V'$.
6: Solve the LP with updated H. Update $\overline{x}$ to be the new LP solution.
7: Update $V' = \{v \in V(H) : \overline{x}_v \geq \gamma\}$. Update $U \leftarrow U \cup V'$.
8: Output U and the updated H.

Let the final updated hypergraph H when Algorithm 1 terminates be denoted by H'. Let the optimal cost of the solution $\overline{x}$ for the vertex cover on H' be denoted by OPT'. We prove that the size of the vertex cover output by the algorithm is not too large:

Lemma 170. *When the above recursive thresholding algorithm (Algorithm 1) terminates, we have* $|U| \leq t' \cdot \big(\mathsf{OPT} - \mathsf{OPT}'\big)$.

Proof. We will inductively prove the following: after line 6 in the while loop of the algorithm, $|U| \leq t' \cdot (\mathsf{OPT} - \mathsf{OPT}_{new})$ where OPT_{new} is the cost of the current optimal solution $\overline{x}$. Let $\overline{x}'$ is the optimal solution before deleting V' from H. Let OPT_{old} be the cost of the solution $\overline{x}'$. By inductive hypothesis, we have $|U| - |V'| \leq t' \cdot (\mathsf{OPT} - \mathsf{OPT}_{old})$.

We claim that $|V'| \leq t' \cdot (\mathsf{OPT}_{old} - \mathsf{OPT}_{new})$. As $\overline{x}$ is an optimal vertex cover of H, we have that $\overline{x}'$ restricted to H has cost at least OPT_{new}. This implies that $\sum_{v \in V'} \overline{x}'_v \geq \mathsf{OPT}_{old} - \mathsf{OPT}_{new}$. As each $\overline{x}'_v, v \in V'$ is at least $\frac{1}{t'}$, we obtain the required claim. $\square$

We are now ready to state our main algorithm for the AHTP. The input to the algorithm is a t-uniform hypergraph G, and the output is a vertex cover for the hypergraph $H = G^{(t-1)}$.

Algorithm 2 Main algorithm

Apply Algorithm 1 to obtain U and let $H' = (V(H'), E(H'))$ be the updated H. Let $\overline{x}$ be an optimal solution of the vertex cover LP on H' with $\overline{x}_v \leq \gamma$ for all $v \in V(H')$.

Let $S \subseteq V(H')$ be defined as $S = \{v : V(H') : \overline{x}_v > 0\}$.

Let $\delta = \sqrt{\frac{4 \ln t}{t-1}}$.

Color the vertices $[n]$ of G using $c : [n] \to \{0, 1\}$ uniformly and independently at random.

For a vertex $v \in S$ and a color $i \in \{0, 1\}$, let $C_i(v)$ denote the number of nodes that are colored with the color i i.e. $\triangleright$ Recall that $S \subseteq V(H') \subseteq \binom{[n]}{t-1}$.

$$C_i(v) = |\{j \in v : c(j) = i\}|$$

Let $S' \subseteq S$ be defined as the set of vertices in S where the discrepancy between two colors is high:

$$S' = \left\{ v \in S : \exists i \in \{0, 1\} : C_i(v) \leq (1 - \delta)\frac{t-1}{2} \right\}$$

We now define a function $f : S \to \{0, 1\}$ as $f(v) = C_1(v) \mod 2$.

For $i \in \{0, 1\}$, let $f^{-1}(i)$ denote the set of all the vertices $v \in S$ such that $f(v) = i$.

Let $p \in \{0, 1\}$ be such that $|f^{-1}(p)| \leq |f^{-1}(1-p)|$.

Let $T \subseteq S$ be defined as $T = S' \cup f^{-1}(p)$.

Output $T \cup U$.

9.3.3 Analysis of the algorithm and proof of Theorem 161

We will first prove that Algorithm 2 indeed outputs a valid vertex cover of H.

Lemma 171. *$T \cup U$ is a vertex cover of H.*

Proof. It suffices to prove that T is a vertex cover of H'.

Consider an arbitrary edge $e = (v_1, v_2, \dots, v_t) \in E(H')$ corresponding to the edge $e(G) = \cup_{j\in[t]} v_j = \{u_1, u_2, \dots, u_t\} \in E(G)$. Since $\overline{x}_v \leq \gamma$ for all $v \in V(H')$, we can deduce that $|e \cap S| \geq \frac{1}{\gamma} = t'$.

Our goal is to show that there exists $j \in [t]$ such that $v_j \in T$. We consider two separate cases:

Case 1: If there is a color $i \in \{0, 1\}$ such that there are at most $(1-\delta)\frac{t-1}{2}$ nodes of color i in $e(G)$, then for all $j \in [t], C_i(v_j) \leq (1-\delta)\frac{t-1}{2}$. Since $e \cap S$ is non-empty, there exists $j \in [t]$ such that $v_j \in S$. By definition of S', this implies that $v_j \in S'$ as well, and thus $e \cap T \neq \phi$.

Case 2: Suppose that in the coloring c, both the colors $0, 1$ occur at least $(1-\delta)\frac{t-1}{2}$ times in e. Let $e' = e \cap S$ and let $k = |e'| \geq t'$. Without loss of generality, let $e' = \{v_1, v_2, \dots, v_k\}$. For every $j \in [k]$, let $v_j = e(G) \setminus \{u_j\}$ for $u_j \in [n]$. First, we claim that $t - k < (1-\delta)\frac{t-1}{2}$. We have

$$\begin{aligned} t - k - (1-\delta)\frac{t-1}{2} &\leq t - t' - (1-\delta)\frac{t-1}{2} = \frac{t}{2} - 2\sqrt{t \ln t} - \left(1 - \sqrt{\frac{4 \ln t}{t-1}}\right)\frac{t-1}{2} \\ &= \frac{1}{2}\left(t - 4\sqrt{t \ln t} - (t-1) + 2\sqrt{(t-1)\ln t}\right) \leq \frac{1}{2}\left(1 - 2\sqrt{t \ln t}\right) < 0 \end{aligned}$$

Since each color occurs at least $(1-\delta)\frac{t-1}{2}$ times in $e(G)$, using the above, we can infer that

$$|\{c(u_1), c(u_2), \dots, c(u_k)\}| \geq 2.$$

We define the value $f(e)$ in the same fashion as we have defined $f(v)$ for $v \in S$: For $i \in \{0, 1\}$, let $C_i(e)$ denote the number of nodes $j \in [t]$ such that $c(u_j) = i$, and let $f(e) = C_1(e) \mod 2$. Using this definition, we get

$$f(v_j) = f(e) - c(u_j) \mod 2 \ \forall j \in [k].$$

As $\{c(u_1), c(u_2), \dots, c(u_k)\} = \{0, 1\}$, we have $\{f(v_1), f(v_2), \dots, f(v_k)\} = \{0, 1\}$ as well. Thus, there exists $j \in [k]$ such that $f(v_j) = p$, which proves that $v_j \in f^{-1}(p) \subseteq T$. $\square$

Note that the expected number of nodes of each color $i \in \{0, 1\}$ in a vertex $v = (u_1, u_2, \dots, u_{t-1}) \in S$ is equal to $\frac{t-1}{2}$. The set S' is the set of vertices of S where there is a color that occurs much fewer than its expected value. We prove that this happens with low probability:

Lemma 172. *The expected cardinality of S' is at most $\frac{2}{t}|S|$.*

Proof. Let $v = (u_1, u_2, \dots, u_{t-1}) \in S$ be an arbitrary vertex in S, where $u_1, u_2, \dots, u_{t-1}$ are elements of $[n]$. For a color $i \in \{0, 1\}$, let the random variable $X(i)$ denote to the number of nodes $j \in [t-1]$ such that $c(u_j) = i$. We can write $X(i) = \sum_{j\in[t-1]} X(i, j)$, where $X(i, j)$ is the indicator random variable of the event that $c(u_j) = i$. We have $\mu = \mathbb{E}[X(i)] = \frac{t-1}{2}$. Using multiplicative Chernoff bound (Lemma 167), we can upper bound the probability that $X(i) \leq (1-\delta)\frac{t-1}{2}$ by

$$\Pr\left(X(i) \leq (1-\delta)\frac{t-1}{2}\right) \leq e^{-\frac{\delta^2(t-1)}{4}}$$

For the choice $\delta = \sqrt{\frac{4\ln t}{t-1}}$, the above probability is at most $\frac{1}{t}$. By applying union bound over the two colors and adding the expectation over all the vertices in S, we obtain the lemma. $\square$

Finally, we bound the expected size of the output of the algorithm:

Lemma 173. *The expected cardinality of $T \cup U$ is at most $\left(\frac{t}{2} + 2\sqrt{t\ln t}\right) \cdot \mathsf{OPT}$.*

Proof. Note that by definition, $|f^{-1}(p)| \leq \frac{|S|}{2}$. We bound the expected size of the output of the algorithm $T \cup U$ as

$$\begin{aligned}
\mathbb{E}[|T \cup U|] \leq \mathbb{E}[|T|] + \mathbb{E}[|U|] \ &\leq \ \mathbb{E}[|S'|] + \frac{1}{2}|S| + \mathbb{E}[|U|] \\
&\leq \left(\frac{1}{2} + \frac{2}{t}\right)|S| + \mathbb{E}[|U|] \quad \text{(Using Lemma 172)} \\
&\leq \left(\frac{t}{2} + 2\right)\mathsf{OPT'} + \mathbb{E}[|U|] \quad \text{(Using Lemma 169)} \\
&\leq \left(\frac{t}{2} + 2\sqrt{t\ln t}\right)\mathsf{OPT} \quad \text{(Using Lemma 170)}\ .
\end{aligned}$$

$\square$

Lemma 171 and Lemma 173 together imply Theorem 161.

9.3.4 $(t, 2)$version of AHTP

An interesting generalization of AHTP is the (t, k)-version, the problem of vertex cover on the k-blown-up hypergraph $H = G^{(k)}$ for a t-uniform hypergraph G, for an arbitrary $1 \leq k < t$. The case of $k = 1$ is the standard vertex cover on t-uniform hypergraphs, and $k = t - 1$ is the AHTP. Note that there is a trivial $\binom{t}{k}$-factor approximation algorithm for this problem as it can be cast as an instance of vertex cover on a $\binom{t}{k}$-uniform hypergraph. The above algorithm can be shown to achieve a $\binom{t}{k}c(k)$ approximation guarantee for the general problem where $c(k) \to \frac{1}{2} + o(1)$ as $k \to t - 1$.

We now turn our attention to the interesting case of $k = 2$. When the hypergraph G consists of t-cliques in a graph, the vertex cover problem on $G^{(2)}$ is the generalization of Tuza's problem where we try to hit all t-cliques with the fewest possible edges. Note that in this case, the trivial hypergraph vertex cover algorithm achieves a $\binom{t}{2}$-factor approximation. We describe how a simplified version of our algorithm can be used to get a $\frac{t^2}{4}$-factor guarantee: Let $H = G^{(2)} = (V(H), E(H))$, and we iteratively solve the Vertex Cover LP on H to round all the vertices with value at least $\frac{4}{t^2}$. In the remaining instance, we let $S \subseteq V(H)$ to be the vertices of H that are assigned non-zero LP value i.e. $S = \{v \in V(H) : x_v > 0\}$. We use a color coding function $c : [n] \to \{0, 1\}$ picked uniformly and independently at random, and we output all the vertices $T = \{\{i, j\} \in S : c(i) = c(j)\}$. The expected size of T is at most $\frac{1}{2}\binom{t}{2}\mathsf{OPT} \leq \frac{t^2}{4}\mathsf{OPT}$ as the cardinality of S is at most $\binom{t}{2}\mathsf{OPT}$.

We now argue that T is indeed a vertex cover of H. Consider an edge $e = \{v_{i,j} : i \neq j \in [t]\} \in E(H)$ corresponding to the edge $e' = (u_1, u_2, \ldots, u_t) \in E(G)$. Recall that every element of e corresponds to a subset of size 2 of e', and thus, without loss of generality, let $v_{i,j} = \{u_i, u_j\}$ for all $i, j \in [t]$. As $x_{v_{i,j}} < \frac{4}{t^2}$ for all $i, j \in [t]$, there are greater than $\frac{t^2}{4}$ pairs of indices i, j such that $x_{v_{i,j}} > 0$, or equivalently, $v_{i,j} \in S$. Thus, $|e \cap S| > \frac{t^2}{4}$. For every function $c : [t] \to \{0, 1\}$, the number of pairs of indices $i \neq j \in [t]$ such that $c(i) \neq c(j)$ is at most $\frac{t^2}{4}$. Thus, there are at least $\binom{t}{2} - \frac{t^2}{4}$ pairs of indices $i \neq j \in [t]$ such that $c(u_i) = c(u_j)$. As $|e \cap S| > \frac{t^2}{4}$, there exists a pair of indices $i \neq j \in [t]$ such that $v_{i,j} \in S$, $c(u_i) = c(u_j)$, which implies that $v_{i,j} \in T$. Thus, for every edge $e \in E(H)$ of H, there exists an element $v \in e$ such that $v \in T$, which completes the proof that T is a vertex cover of H.

As a corollary of the algorithm, we deduce that for all hypergraphs $H = G^{(2)}$ of a t-uniform hypergraph G,

$$\tau(H) \leq \frac{t^2}{4}\tau^*(H)$$

This proves the fractional version of a conjecture due to Aharoni and Zerbib [AZ20] (Conjecture 1.4) for the case when $k = 2$.

9.4 Forbidden sub-hypergraphs and Tuza's conjecture

Since AHTP is the problem of vertex cover on $H = G^{(t-1)}$ for a given t-uniform hypergraph G, an interesting question is to characterize the t-uniform hypergraphs H that can arise as the blown-up hypergraph $G^{(t-1)}$ of some t-uniform hypergraph G. A very simple necessary condition is that the hypergraph H should be simple. However, this is not sufficient—there are simple t-uniform hypergraphs H that cannot be written as $H = G^{(t-1)}$ for any G. For example, the t-tent hypergraph (Definition 174) is a simple hypergraph, but cannot be written as a $(t-1)$-blown-up hypergraph. A natural question in this context is the following:

> Is there a finite set of hypergraphs $\mathcal{F}$ such that every hypergraph that does not have any member of $\mathcal{F}$ as a sub-hypergraph can be represented as $G^{(t-1)}$ for some t-uniform hypergraph G?

In addition to its inherent structural interest, the above question can shed light on Tuza's conjecture. Recall that Aharoni and Zerbib[AZ20] proposed a generalization of Tuza's conjecture stating that $\tau\left(G^{(2)}\right) \leq 2 \cdot \nu\left(G^{(2)}\right)$ for all 3-uniform hypergraphs G. They suggested that understanding the structure of blown-up hypergraphs, and specifically, the sub-hypergraphs that it excludes might be a promising approach to establish this conjecture. In particular, they observed that the blown-up hypergraphs do not contain "tents" as a sub-hypergraph.

Definition 174. *A t-tent (Figure 9.1) is a set of four t-uniform edges e_1, e_2, e_3, e_4 such that*

1. $\cap_{i=1}^{3} e_i \neq \phi$.
2. $|e_4 \cap e_i| = 1$ *for all* $i \in [3]$.
3. $e_4 \cap e_i \neq e_4 \cap e_j$ *for all* $i \neq j \in [3]$.

In [AZ20], the authors pose the following question. Note that an answer in the affirmative would resolve Tuza's conjecture, and in fact its above generalization that $\tau\left(G^{(2)}\right) \leq 2\nu\left(G^{(2)}\right)$ for all 3-uniform hypergraphs G.

Problem 175. *Is it true that for every 3-uniform hypergraph H without a 3-tent, $\tau(H) \leq 2 \cdot \nu(H)$?*

We answer this question in the negative. In fact, we prove a stronger statement that there can be no forbidden substructure-based Tuza's theorem.

Theorem 176. *Let $\mathcal{F} = \{F_1, F_2, \ldots, F_\ell\}$ be an arbitrary set of t-uniform hypergraphs such that for every t-uniform hypergraph G, the blown-up hypergraph $G^{(t-1)}$ does not contain any $F_i \in \mathcal{F}$ as a sub-hypergraph.*

Then, for every $\epsilon > 0$, there exists a hypergraph H' that does not contain any member of $\mathcal{F}$ as a sub-hypergraph and which satisfies $\tau(H') \geq (t-\epsilon)\tau^*(H') \geq (t-\epsilon)\nu(H')$.

By setting $\mathcal{F}$ to be the single 3-tent hypergraph, we obtain a counterexample to Problem 175. Furthermore, when $t = 3$, the construction we give to prove Theorem 176 will belong to the class of 3-uniform hypergraphs H obtained from a given graph G with the vertex set of H being the edge set of G, and every triangle in G forming an edge in H. Thus, there is no "local" proof of Tuza's conjecture that uses only substructure properties of the underlying hypergraph.

We call a hypergraph non-trivial if it has at least two edges. Before we prove the above theorem, we use a definition from [FM08].

Definition 177. *Let F be a non-trivial t-uniform hypergraph. Then,*

$$\rho(F) = \max_{F' \subseteq F} \frac{e'-1}{v'-t}$$

where F' is a non-trivial subhypergraph of F with $e' > 1$ edges and v' vertices.

We now return to the proof of Theorem 176.

Proof. (of Theorem 176) We will first prove that $\rho(F_i) > \frac{1}{t-1}$ for all $i \in [\ell]$. Suppose for contradiction that there exists a t-uniform hypergraph $F_i \in \mathcal{F}$ such that $\rho(F_i) \leq \frac{1}{t-1}$. Without loss of generality, we can assume that F_i is connected. Order the edges of F_i as $\{e_1, e_2, \ldots, e_m\}$ such that for every $j > 1$, $e_j \cap (e_1 \cup e_2 \cup \ldots \cup e_{j-1}) \neq \phi$. For every $j \geq 1$, let F'_j be the subhypergraph induced by $\{e_1, e_2, \ldots, e_j\}$. As $\rho(F_i) \leq \frac{1}{t-1}$, we can infer that for every $j > 1$,

$$\frac{|E(F'_j)| - 1}{|V(F'_j)| - t} \leq \frac{1}{t-1}$$

which implies that $|V(F'_j)| \geq (t-1)|E(F'_j)| + 1 = (t-1)j + 1$. As $|V(F'_j)| = |V(F'_{j-1})| + t - |e_j \cap (e_1 \cup e_2 \cup \ldots \cup e_{j-1})|$, we get that $|V(F'_j)| \leq |V(F'_{j-1})| + t - 1$, which combined with the above shows that the inequality is in fact tight for every $j > 1$. Thus, for every $j > 1, |V(F'_j)| = |V(F'_{j-1})| + t - 1$, which implies that for every $j > 1$,

$$|e_j \cap (e_1 \cup e_2 \cup \ldots \cup e_{j-1})| = 1.$$

We now construct a t-uniform hypergraph H such that F_i is isomorphic to $H^{(t-1)}$. We construct the hypergraph H inductively via $H_1, H_2, \ldots, H_m = H$ such that $H_j^{(t-1)}$ is isomorphic to F'_j for all $j \in [m]$. First, we set the hypergraph H_1 to be equal to the t-uniform hypergraph on t vertices with a single edge. H_1 is trivially isomorphic to F'_1. Assume by inductive hypothesis that there is a hypergraph H_k such that F'_k is isomorphic to $H_k^{(t-1)}$ for some $k \in [m-1]$. Let $\phi : F'_k \to H_k^{(t-1)}$ be the isomorphism between the two hypergraphs. The hypergraph F'_{k+1} is obtained from F'_k by adding an edge e_{k+1} such that e_{k+1} intersects with F'_k in exactly one vertex $v \in V(F'_k)$. Recall that the vertex set of $H_k^{(t-1)}$ is the set of subsets of vertices of H_k of size $t-1$. Thus, $\phi(v) = \{(p_1, p_2, \ldots, p_{t-1})\}$ for a set of vertices $p_1, p_2, \ldots, p_{t-1} \in V(H_k)$. We construct H_{k+1} by introducing a new vertex v' and adding the edge $\{v', p_1, p_2, \ldots, p_{t-1}\}$ to the hypergraph H_k. Thus, $H_{k+1}^{(t-1)}$ is obtained from $H_k^{(t-1)}$ by adding single edge that intersects with $H_k^{(t-1)}$ at exactly one vertex, that is $\{p_1, p_2, \ldots, p_{t-1}\}$. Hence, $H_{k+1}^{(t-1)}$ is isomorphic to F'_{k+1}, completing the proof. This proves that there exists a t-uniform hypergraph $H = H_m$ such that $F_i = H^{(t-1)}$, contradicting the fact that no $(t-1)$-blown-up hypergraph contains F_i as a subhypergraph.

Thus, $\rho(F_i) > \frac{1}{t-1}$ for all $i \in [\ell]$. Let $\rho = \min_{i \in [\ell]} \rho(F_i) > \frac{1}{t-1}$. Consider a random t-uniform hypergraph H' on n vertices sampled by picking each edge independently with probability $p = n^{-\frac{1}{\rho}}$. We now delete the edges in a maximal collection of edge disjoint copies of members of $\mathcal{F}$ from H'. It has been proved [BFM10; FM08] that the maximum independent set $\alpha(H')$ of this construction satisfies

$$\alpha(H') \leq \tilde{O}\left(n^{\frac{1}{(t-1)\rho}}\right)$$

with high probability. Thus, there exists a t-uniform hypergraph H' with n vertices without any substructure from $\mathcal{F}$ such that $\tau(H') \geq (1 - o(1))n$. Since for any t-uniform hypergraph H' on n vertices, $\nu(H') \leq \tau^*(H') \leq \frac{1}{t}n$, this proves the claimed factor $(t - \epsilon)$ gap between $\tau(H')$ and $\tau^*(H')$ for every positive constant $\epsilon > 0$. □

9.4.1 Explicit construction of tent-free hypergraphs

We now describe an explicit hypergraph giving negative answer to Problem 175. Our counterexample is a hypergraph with vertex set $[3]^n$ for large enough n and the edge set is the set of all combinatorial lines that we formally define below:

Definition 178. *(Combinatorial lines in $[3]^n$) A set of three distinct vectors $u = (u_1, u_2, \ldots, u_n), v = (v_1, v_2, \ldots, v_n), w = (w_1, w_2, \ldots, w_n) \in [3]^n$ forms a combinatorial line if there exists a subset $S \subseteq [n]$ such that*

1. *For all $i \in [n] \setminus S$, $u_i = v_i = w_i$.*
2. *There exist three distinct integers $u', v', w' \in [3]$ such that for all $i \in S$, $u_i = u', v_i = v', w_i = w'$.*

We will use the following seminal result about combinatorial lines:

Theorem 179. *(Density Hales Jewett Theorem [FK91],[Pol12]) For every positive integer k and every real number $\delta > 0$ there exists a positive integer $\mathsf{DHJ}(k, \delta)$ such that if $n \geq \mathsf{DHJ}(k, \delta)$ and A is any subset of $[k]^n$ of density at least δ, then A contains a combinatorial line.*

We now give a proof of Theorem 163.

Proof. The hypergraph that we use $H = (V, E)$ has $V = [3]^n$ for n large enough to be set later, and the edges are all the combinatorial lines in $[3]^n$. First, we claim that the above defined hypergraph does not have a 3-tent. Suppose for contradiction that there are edges e_1, e_2, e_3, e_4 satisfying the properties of Definition 174. Let $u = (u_1, u_2, \ldots, u_n) \in e_4 \cap e_1, v = (v_1, v_2, \ldots, v_n) \in e_4 \cap e_2, w = (w_1, w_2, \ldots, w_n) \in e_4 \cap e_3$. Note that $e_4 = \{u, v, w\}$. Thus, there exists a subset $S \subseteq [n]$ such that for all $i \in [n] \setminus S$, $u_i = v_i = w_i$. Without loss of generality, we can also assume that for all $i \in S$, $u_i = 1, v_i = 2, w_i = 3$.

Let $x = (x_1, x_2, \ldots, x_n) \in e_1 \cap e_2 \cap e_3$. Note that $\{x, u\} \subseteq e_1, \{x, v\} \subseteq e_2, \{x, w\} \subseteq e_3$. Consider an arbitrary element $p \in S$, and without loss of generality, let $x_p = 1$. Thus, we have that $x_p = 1, v_p = 2$ and both x, v share the combinatorial line e_2. This implies that there exist a subset $S_2 \subseteq [n]$ such that for all $i \in [n] \setminus S_2$, $x_i = v_i$ and for all $i \in S_2$, $x_i = 1, v_i = 2$. Similarly, there exists a subset $S_3 \subseteq [n]$ such that for all $i \in [n] \setminus S_3$, $x_i = w_i$ and for all $i \in S_3$, $x_i = 1, w_i = 3$.

Note that $S_2 \subseteq S$. Suppose for contradiction that there exists $j \in S_2 \setminus S$. Then, we have $v_j = 2, x_j = 1$. However, since $v_i = w_i$ for all $i \in [n] \setminus S$, we get that $w_j = 2$, and thus, $j \notin S_3$, which implies that $x_j = w_j = 2$, a contradiction. Thus, $S_2 \subseteq S$, and similarly $S_3 \subseteq S$. We can also observe that $S_2 \neq S$ since in that case, $x = u$ which cannot happen since $|e_4 \cap e_2| = 1$. By the same argument on e_3, we can deduce that $S_3 \neq S$. As S_2 is a strict subset of S, there exists $j \in S \setminus S_2$. As $v_i = x_i$ for all $i \in [n] \setminus S_2$, $x_j = v_j = 2$. As $j \in S$, we have $w_j = 3$. However, as $w_j \neq x_j$, this implies that $j \in S_3$, which then implies that $x_j = 1$, a contradiction.

Now, we will prove that for large enough n, $\tau(H) > (3 - \epsilon)\nu(H)$. Let $N = 3^n$. Since the cardinality of V is equal to N, we have $\nu(H) \leq \frac{N}{3}$. We apply Theorem 179 with $k = 3, \delta = \frac{\epsilon}{3}$, and set $n \geq \mathsf{DHJ}(k, \delta)$. Thus, we can infer that in any subset $T \subseteq V$ of size $\frac{\epsilon}{3}N$, there exists an edge of H fully contained in T. Thus, we get that $\tau(H) > (1 - \frac{\epsilon}{3})N$, which gives $\tau(H) > (3 - \epsilon)\nu(H)$. □

9.5 Vertex cover and set cover on simple hypergraphs

As mentioned earlier, the edges in a $(t-1)$-blown-up hypergraph of a t-uniform hypergraph can intersect on at most one element, so such hypergraphs are simple. In this section, we will take a step back and address to what extent improved approximation algorithms are possible for vertex cover on simple hypergraphs. We will also consider the dual problem, of covering the vertices by the fewest possible hyperedges, namely the set cover problem, on simple hypergraphs but without any restriction on the size of the hyperedges. Note that a hypergraph is simple if and only if the edge-vertex incidence bipartite graph does not contain a copy of $K_{2,2}$. Thus, a hypergraph is simple if and only if its dual is simple.

9.5.1 Vertex cover on simple t-uniform hypergraphs

We now prove Theorem 164 which shows that simple hypergraphs are still rich enough to preclude a non-trivial approximation to vertex cover. Our hardness is established using a reduction from the general problem of vertex cover on t-uniform hypergraphs. In particular, we use the following result:

Theorem 180. *([Din+05]) For every constant $\epsilon > 0$ and $t \geq 3$, the following holds: Given a t-uniform hypergraph $G = (V, E)$, it is NP-hard to distinguish between the following cases:*

1. *Completeness: G has a vertex cover of measure $\frac{1+\epsilon}{t-1}$.*
2. *Soundness: Any subset of V of measure ϵ contains an edge from E.*

We give a randomized reduction from Theorem 180 to Theorem 164. The approach is similar to the one used in [GL17] for showing the inapproximability of H-Transversal in graphs. It was also used in the recent tight hardness for Max Coverage on simple set systems [CKL21].

Let us instantiate Theorem 180 with ϵ replaced by $\epsilon' = \frac{\epsilon}{4}$, and let the resulting hypergraph be denoted by G. Now, given this t-uniform hypergraph $G = (V, E)$, we output a t-uniform hypergraph $H = (V', E')$ as follows: Let $n = |V|, m = |E|$. We have integer parameters B, P depending on ϵ, t, n, m to be set later. The vertex set of H is $V' = V \times [B]$–we have a cloud of B vertices $v^1, v^2, \ldots, v^B$ in V' corresponding to every vertex $v \in V$. For every edge $e = (v_1, v_2, \ldots, v_t) \in E$, we pick P edges $e^1, e^2, \ldots, e^P$ with $e^i = ((v_1)_i, (v_2)_i, \ldots, (v_t)_i)$ and add them to E', where for each $j \in [t]$ and $i \in [P]$, $(v_j)_i$ is chosen uniformly and independently at random from $(v_j)^1, (v_j)^2, \ldots, (v_j)^B$. Thus, so far, we have added mP edges to E'.

We first upper bound the expected value of the number of pairs of edges in E' that intersect in more than one vertex. Order the edges in E' as $e_1, e_2, \ldots, e_{mP}$. Let X denote the random variable that counts the number of pairs of edges in E' that intersect in more than one vertex. For every pair of indices $i, j \in [mP]$, let the random variable X_{ij} be the indicator variable of the event that the edges e_i and e_j of E' intersect in greater than one vertex. Note that the edges in E corresponding to e_i and e_j have at most t vertices in common. Thus, the probability that e_i and e_j intersect in at least two vertices is upper bounded by $\binom{t}{2}\frac{1}{B^2}$. Summing over all the pairs i, j, we get

$$\mathbb{E}[X] \leq \binom{mP}{2}\binom{t}{2}\frac{1}{B^2} \leq \frac{m^2 t^2 P^2}{B^2}.$$

By Markov's inequality, with probability at least $\frac{9}{10}$, X is at most $\frac{10m^2t^2P^2}{B^2}$.

We consider all the pairs of edges that intersect in more than one vertex in E', and arbitrarily delete one of those edges. Let the resulting set of edges be denoted by E''. The final hypergraph resulting in this reduction is $H = (V', E'')$. Note that H is indeed a simple hypergraph. We will prove the following:

1. (Completeness) If G has a vertex cover of measure μ, then there is a vertex cover of measure μ in H.
2. (Soundness) If every subset of V of measure ϵ' contains an edge from E, then with probability at least $\frac{4}{5}$, every subset of V' of measure ϵ contains an edge from E''.

Completeness. If G has a vertex cover of size μn, then picking all the vertices in V' in the cloud corresponding to these vertices ensures that H has a vertex cover of size $\mu n B$. Thus, in the completeness case, there is a vertex cover of measure μ in H.

Soundness. Suppose that every ϵ' measure subset of V contains an edge from E. Our goal is to show that with probability at least $\frac{4}{5}$, every ϵ measure subset of V' contains an edge from E''. We first prove the following lemma:

Lemma 181. *With probability at least $\frac{9}{10}$ over the choice of E', the following holds: For every edge $e = (v_1, v_2, \ldots, v_t) \in E$, and every subset $S \subseteq V'$ such that for each $i \in [t]$, S contains at least $\frac{\epsilon}{4}B$ vertices from $\{v_i^1, v_i^2, \ldots, v_i^B\}$, there exists an edge $e' \in E'$ all of whose vertices are in S.*

Proof. The probability that there exists an edge $e = (v_1, v_2, \ldots, v_t)$ and a subset S which contains at least $\frac{\epsilon}{4}B$ vertices from each cloud and does not contain any edge from E' is at most

$$m2^{tB}\left(1 - \left(\frac{\epsilon}{4}\right)^t\right)^P \leq m2^{tB - \log e \frac{\epsilon^t P}{4^t}} \leq \frac{1}{10}$$

when $P = maB$ where $a := a(t, \epsilon) = \frac{4^{t+2}t}{\epsilon^t}$. □

Using the above lemma, we can conclude that with probability at least $\frac{4}{5}$, $X \leq \frac{10m^2t^2P^2}{B^2} = 10m^4t^2a^2$ and for every edge $e \in E$ and every subset $S \subseteq V'$ such that for each $i \in [t]$, S contains at least $\frac{\epsilon}{4}B$ vertices from $\{v_i^1, v_i^2, \ldots, v_i^B\}$, there exists an edge $e' \in E'$ all of whose vertices are in S. We claim that this implies that with probability at least $\frac{4}{5}$, every ϵ measure subset of V' contains an edge of E''. Consider an arbitrary subset $U \subseteq V'$ such that $|U| \geq \epsilon nB$. We choose B large enough such that $t(10m^4t^2a^2) \leq \frac{\epsilon}{2}nB$. Thus, the set of the vertices W in the edges deleted from E' to obtain E'' has cardinality at most $\frac{\epsilon}{2}nB$.

Let $U' = U \setminus W$. Note that all the edges in U' that are in E' are present in E'' as well. As U' has a measure of at least $\frac{\epsilon}{2}$ in V', for at least $\frac{\epsilon}{4}n$ vertices v in V, U' should contain at least $\frac{\epsilon}{4}$ fraction of the vertices in the cloud $\{v^1, v^2, \ldots, v^B\}$. Since otherwise, the cardinality of U' is at most $\left(n - \frac{\epsilon n}{4}\right) \cdot \frac{\epsilon B}{4} + \frac{\epsilon n}{4} \cdot B < \frac{\epsilon nB}{2}$, a contradiction. By Lemma 181, we can deduce that there exists an edge $e \in E'$ all of whose vertices are in U', which implies that the edge e is in E'' as well. This proves that in the soundness case, with probability at least $\frac{4}{5}$, there exists an edge in every ϵ measure subset of V'.

This completes the proof of Theorem 164. Under the Unique Games Conjecture [Kho02a], the hardness of vertex cover in t-uniform hypergraphs can be improved to $t - \epsilon$. We remark that we can get the same hardness for simple hypergraphs by our reduction.

9.5.2 Set Cover on Simple Set Systems

In the set cover problem, there is a set family $\mathcal{S} \subseteq 2^X$ on a universe $X = [n]$, and the goal is to cover the universe $[n]$ with as few sets from the family as possible. The greedy algorithm where we repeatedly pick the set that covers the maximum number of new elements achieves a $\ln n$-factor approximation algorithm for the problem, and this is known to be optimal. We consider the same problem under the restriction that the family $\mathcal{S}$ is a simple set system i.e. for every $i \neq j$, $|S_i \cap S_j| \leq 1$. Surprisingly, in contrast with the hardness result for vertex cover, simplicity of the set family helps in achieving better approximation factor for the set cover problem.

Theorem 182. *(Theorem 165 restated) The greedy algorithm achieves a* $\left(\frac{\ln n}{2}+1\right)$*-approximation guarantee for the set cover problem on simple set systems over a universe of size* n*. Furthermore, the bound is essentially tight for the greedy algorithm—there is a simple set system on which the approximation factor of greedy exceeds* $(\frac{\ln n}{2}-1)$.

Proof. First, we prove the upper bound. Let the optimal solution size be equal to k i.e. there is $\mathcal{T} = \{S_1, S_2, \ldots, S_k\} \subseteq \mathcal{S}$ such that the union of sets in $\mathcal{T}$ is equal to $[n]$. For every set $S \in \mathcal{S} \setminus \mathcal{T}$, $|S \cap S_i| \leq 1$ by the simplicity of the set system, and thus, we get that

$$\forall S \in \mathcal{S} \setminus \mathcal{T}, |S| \leq k. \tag{9.1}$$

We now consider two different cases:

1. Suppose that $k \geq \sqrt{n}$. We recall that the greedy algorithm in fact achieves a $\log |S_{max}|$-factor approximation algorithm for set cover on general instances where $|S_{max}|$ is the size of the largest set in the family. Thus, after the greedy algorithm picks t sets each of which cover at least $\sqrt{n}$ new elements, in the remaining instance, we have $|S_{max}| \leq \sqrt{n}$. As there are k sets that cover the remainining instance, the greedy algorithm picks at most $k\frac{\ln n}{2}$ sets after picking the t sets. As each of the t sets cover at least $\sqrt{n}$ new elements, $t \leq \sqrt{n}$. Overall, the total number of sets used by the greedy algorithm is equal to

$$t + k\frac{\ln n}{2} \leq \sqrt{n} + k\frac{\ln n}{2} \leq \left(1 + \frac{\ln n}{2}\right) k$$

2. Suppose that $k < \sqrt{n}$. In this case, using (9.1), we can infer that there are at most k sets with size at least k in the family. Thus, after the greedy algorithm picks k sets, in the remaining instance, each set has size at most k, and thus, greedy algorithm picks at most $k \ln k$ sets. Overall, the total number of sets picked by the greedy algorithm is equal to

$$k + k \ln k \leq k + k\frac{\ln n}{2} = \left(1 + \frac{\ln n}{2}\right) k$$

Thus, in both the cases, the greedy algorithm picks at most $k\left(1 + \frac{\ln n}{2}\right)$ sets.

A hard instance for the greedy algorithm. We now prove that the above bound is tight for the greedy algorithm. Fix a large integer k, and let $n = k^2$, $X = [n]$. We first add k sets to the family $\mathcal{S}$ $S_1, S_2, \ldots, S_k$ where $S_j = \{(j-1)k+1, (j-1)k+2, \ldots, jk\}$. Note that these k sets together cover the whole universe X. We view the universe X as k blocks, with the j'th block comprising of the set S_j.

We now add $m = (k-1)\ln k$ additional pairwise disjoint sets $T_1, T_2, \ldots, T_m$ to $\mathcal{S}$ such that the greedy algorithm picks the set T_i in the i'th iteration. We choose the sets $T_i, i \in [m]$, as follows:

1. For $j \in [k]$, let the set X_j be the uncovered elements of the block S_j. Let $a_j = |X_j|$ for all $j \in [k]$. We initially set $X_j = S_j$ for all $j \in [k]$.
2. At every iteration $i \in [m]$:

(a) Sort the elements $\{a_1, a_2, \ldots, a_k\}$ such that $a_{\alpha_1} \geq a_{\alpha_2} \geq \ldots \geq a_{\alpha_k}$ where $(\alpha_1, \alpha_2, \ldots, \alpha_k)$ is a permutation of $[k]$. Let $p = a_{\alpha_1} \leq k$ be the largest number of uncovered elements in a block.

(b) Let $P = \{\alpha_1, \alpha_2, \ldots, \alpha_p\}$. For $l \in [p]$, let $u_l \in [n]$ be equal to the largest element in X_{α_l}.

$$u_l = \max\{b \mid b \in X_{\alpha_l}\}$$

We set

$$T_i = \{u_l \mid l \in [p]\}$$

Furthermore, we set $X_{\alpha_l} = X_{\alpha_l} \setminus u_l$ for all $l \in [p]$. We also update $a_j, j \in [k]$ as $a_j = |X_j|$ for all $j \in [k]$.

In the above procedure to output the sets $T_i, i \in [m]$, the cardinality of $|T_i| \geq |T_{i+1}|$ for all i. Furthermore, in the ith iteration of the above procedure, the cardinality of T_i is at least the number of elements in any block that are not covered yet. This ensures that the greedy algorithm in the ith iteration picks the set T_i. Furthermore, as the sets T_is are all mutually disjoint, and intersect each block at most once, the resulting set system is indeed a simple set system.

Our goal is to prove that after all the m sets are picked, there are still uncovered elements in $[n]$. For an integer $i \in [m]$, we let $s_i \in [n]$ denote the number of elements not covered by the greedy algorithm before the set T_i is picked. For $i \in [m]$, the size of the set T_i picked by the greedy algorithm in the ith iteration is equal to the largest number of uncovered elements in a block i.e. the value of a_{α_1} in the ith iteration. Based on the updating procedure followed above, we can infer that this value is equal to $|T_i| = \left\lceil \frac{s_i}{k} \right\rceil$. This follows from the fact that at any iteration of the above procedure, the sorted values a_{α_1} and a_{α_k} satisfy $a_{\alpha_1} \leq a_{\alpha_k} + 1$.

We have $s_1 = n = k^2$, and

$$s_{i+1} = s_i - |T_i| = s_i - \left\lceil \frac{s_i}{k} \right\rceil \geq s_i \left(1 - \frac{1}{k}\right) - 1$$

By setting $t_i = s_i + k$ for $i \in [m]$, we get

$$t_{i+1} \geq t_i \left(1 - \frac{1}{k}\right)$$

Thus, we get

$$\begin{aligned}
t_{m+1} &\geq t_1 \left(1 - \frac{1}{k}\right)^m \\
&= (k^2 + k) \left(1 - \frac{1}{k}\right)^{(k-1)\ln k} \\
&\geq (k^2 + k) \exp\left(-\frac{\frac{1}{k}}{1 - \frac{1}{k}}(k-1)\ln k\right) \quad \text{(Using } 1 - x \geq e^{\frac{-x}{1-x}}\ \forall\, 0 \leq x < 1\text{)} \\
&= k + 1
\end{aligned}$$

Thus, $s_{m+1} \geq 1$, which proves that there are elements that are not covered after the greedy algorithm uses $m = (k-1)\ln k$ sets. This completes the proof that there are simple set systems on n elements with $k = \sqrt{n}$ sets covering all the elements where as the greedy algorithm picks at least $(k-1)\ln k \geq k\left(\frac{\ln n}{2} - 1\right)$ sets.

$\square$

Chapter 10

Scheduling with non-uniform communication delay

10.1 Introduction

We study the problem of scheduling jobs with precedence and non-uniform communication delay constraints on identical machines to minimize the makespan objective function. This classic model was first introduced by Rayward-Smith [Ray87] and Papadimitriou and Yannakakis [PY90]. In this problem, we are given a set J of n jobs, where each job j has a processing length $p_j \in \mathbb{Z}_+$. The jobs need to be scheduled on m identical machines. The jobs have precedence and communication delay constraints, which are given by a partial order $\prec$. A constraint $j \prec j'$ encodes that job j' can only start after job j is completed. Moreover, if $j \prec j'$ and j, j' are scheduled on different machines, then j' can only start executing at least $c_{jj'}$ time units after j had finished. On the other hand, if j and j' are scheduled on the same machine, then j' can start executing immediately after j finishes. The goal is to schedule jobs *non-preemptively* to minimize the makespan objective function, which is defined as the completion time of the last job. In a non-preemptive schedule, each job j needs to be assigned to a single machine i and executed during a contiguous time interval of length p_j. In the classical scheduling notation, the problem is denoted by $\mathsf{P} \mid \mathsf{prec}, c_{jk} \mid C_{\mathsf{max}}$.[1] A closely related problem is $\mathsf{P}\infty \mid \mathsf{prec}, c_{jk} \mid C_{\mathsf{max}}$, where the scheduler has access to an unbounded number of machines.

Scheduling jobs with precedence and communication delays has been studied extensively over many years [Ray87; PY90; MK97; HM01; TY92; HLV94; Gir+08]. Furthermore, due to its relevance in datacenter scheduling problems and large-scale training of ML models, there has been a renewed interest in more applied communities; we refer the readers to [Cho+11; Guo+12; HCG12; Shy+18; Zha+12; Zha+15; Luo+16; Nar+19; Mir+17; GCL18; JZA19; Tar+20]. However, from a theoretical standpoint, besides NP-hardness results, very little was known in

[1]We adopt the convention of [Gra+79; VLL90], where the respective fields denote: **(1) machine environment:** Q for related machines, P for identical machines, **(2) job properties:** prec for precedence constraints; c_{jk} for communication delays; c when all the communication delays are equal to c; $p_j = 1$ for the unit-length case, **(3) objective:** C_{max} for minimizing makespan.

terms of the algorithms for the problem until the recent work by Maiti et al. [Mai+20] and Davies et al. [Dav+20; Dav+21]. These very recent papers designed polylogarithmic approximation algorithms for the special case when all the communication delays are *equal*. We survey these results in Section 10.1.2. In fact, the problems of scheduling jobs with communication delays are some of the well-known open questions in approximation algorithms and scheduling theory, and have resisted progress for a long time. For this reason, the influential survey by Schuurman and Woeginger [SW99b] and its recent update by Bansal [Ban17] list understanding the approximability of the problems in this space as one of the top-10 open questions in scheduling theory.

In particular, an open problem in these surveys asks if the non-uniform communication delay problem on identical machines, even assuming an unbounded number of machines ($\mathsf{P}\infty \mid \mathsf{prec}, c_{jk} \mid C_{\mathsf{max}}$), admits a constant-factor approximation algorithm. We answer this question in the negative.

Theorem 183. *For every constant $\epsilon > 0$, assuming $\mathsf{NP} \not\subseteq \mathsf{ZTIME}\left(n^{(\log n)^{O(1)}}\right)$, the non-uniform communication delay problem ($\mathsf{P}\infty \mid \mathsf{prec}, c_{jk} \mid C_{\mathsf{max}}$) does not admit a polynomial-time $(\log n)^{1-\epsilon}$-approximation algorithm.*

We remark that our hard instances contain only two distinct values of communication delays (essentially 0 and ∞). Furthermore, as $\mathsf{P}\infty \mid \mathsf{prec}, c_{jk} \mid C_{\mathsf{max}}$, the problem with an unbounded number of machines, is a special case of $\mathsf{P} \mid \mathsf{prec}, c_{jk} \mid C_{\mathsf{max}}$, where the number of machines is specified, our theorem also implies the same hardness for $\mathsf{P} \mid \mathsf{prec}, c_{jk} \mid C_{\mathsf{max}}$.

10.1.1 Our Techniques

Our hardness result is obtained using a reduction via a problem we call Unique Machines Precedence constraints Scheduling (UMPS). In this problem, there are m machines and n jobs $j_1, j_2, \ldots, j_n$ with precedence relations between them. Each job j_l has length $p(l)$ and can be scheduled only on a *unique* machine $M(l) \in [m]$. The objective is to schedule the jobs non-preemptively on the corresponding unique machines, respecting the precedence relations, so as to minimize the makespan objective function. Our proof of Theorem 183 proceeds via two steps:

1. First we show a reduction from an instance I of the UMPS problem to an instance I' of the non-uniform communication delay problem. The key step is to make sure that the set of jobs $J(i)$ that need to be scheduled on machine i in I do not get scheduled on multiple machines in I'. We achieve this by introducing a dummy job j_i^* and introducing precedence constraints from all jobs in $J(i)$ to j_i^* and a very large communication delay. This ensures that $J(i)$ and j_i^* are scheduled on the same machine in I', although this machine need not be i. Despite this, we show that any valid schedule of I' can be mapped back to a feasible schedule of I, with almost the same makespan. Our reduction creates only two types of communication delays and works for the unit-length case.

2. Next we observe that the UMPS problem generalizes the classical job shops problem (see e.g. [Law+93; LMR94; MS11]), whose approximation is well understood [SSW94; CS00; Gol+01; FS02]. The logarithmic hardness result for the acyclic job shops problem by

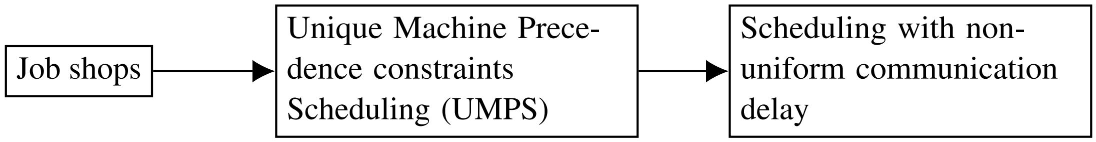

Figure 10.1: Role of the UMPS problem in our hardness reduction.

Mastrolilli and Svensson [MS11] implies a logarithmic hardness of the UMPS problem. We remark that the hardness result of [MS11] only works when jobs have lengths, and hence our Theorem 183 only applies to the setting where jobs have lengths.

In hindsight, our proof of Theorem 183 is surprisingly simple. However, the main conceptual contribution of our proof is in identifying the UMPS problem as a central problem that has implications for the hardness of various scheduling problems. Furthermore, the UMPS problem, which can be viewed as a generalization of the job shop scheduling model or as a highly restricted version of multidimensional scheduling with precedences, or as a restricted assignment problem with precedence constraints, is a fundamental problem to study on its own, both from a theoretical perspective and also from a practical point of view. We believe the UMPS problem is a key intermediate step towards resolving several long-standing open problems in scheduling theory. We make the following two conjectures regarding the approximability of UMPS.

Conjecture 184. *There exists a constant $\epsilon < 1$ such that it is NP-hard to approximate UMPS within a factor of n^ϵ, even when all jobs have unit lengths, where n is the number of jobs.*

Conjecture 185. *There exists an absolute constant $C \geq 1$ such that the following holds. For every constant $\epsilon > 0$, it is NP-hard to approximate UMPS within a $(\log n)^{1-\epsilon}$-factor, even when the number of machines m is at most $(\log n)^C$ and all the jobs have unit lengths, where n is the number of jobs.*

Our second main contribution is to show that the above conjectures imply hardness results for various problems. In particular, Conjecture 185 implies logarithmic hardness for scheduling with precedences on related machines, another top-10 problem in scheduling theory [SW99b; Ban17] and in the approximation algorithm book of Shmoys and Williamson [WS11].

Theorem 186. *Assuming Conjecture 185 and $\mathsf{NP} \not\subseteq \mathsf{DTIME}\left(n^{(\log n)^{O(1)}}\right)$, there exists an absolute constant $\gamma > 0$ such that the problem of scheduling related machines with precedences $(\mathsf{Q} \mid \mathsf{prec} \mid C_{\max})$ has no polynomial-time $O\left((\log m)^\gamma\right)$-factor approximation algorithm.*

Previously, Bazzi and Norouzi-Fard [BN15] introduced a k-partite hypergraph partition problem whose hardness implies a superconstant hardness for scheduling with precedences on related machines. Our reduction uses the same idea of job replication as [BN15], while our soundness analysis is technically more involved. We also show that the hypothesis of [BN15] implies a superconstant hardness of the UMPS problem. Thus, our problem can be viewed as a weaker version of the hypothesis of [BN15] with the same implication towards the hardness of related machines. Furthermore, stronger hardness of the UMPS problem implies better (almost optimal) hardness results for the related machines scheduling problem.

Finally, we note that Conjecture 184 implies that precedence-constrained scheduling (even without communication delays) is very hard to approximate when generalized to the restricted assignment setting or unrelated machines.

Our confidence in the above conjectures stems from the fact that existing techniques, both the classical jobshops algorithms [LMR94] and the recent LP-hierarchies-based algorithms [Mai+20; Dav+20] fail to give non-trivial approximation guarantees for the UMPS problem. Furthermore, a candidate hard instance for the problem is a *layered* instance, where there are precedences between jobs $j_1 \prec j_2$ only if j_1 can be scheduled on the machine i and j_2 can be scheduled on the machine $i+1$. These layered instances are closely related to the k-partite partitioning hypothesis of [BN15] and the integrality gap instances [Mai+20] for the problem of scheduling with uniform communication delays.

10.1.2 A Brief History of the Communication Delay Problem

In this subsection, we give a brief overview of the literature on the problem of scheduling with communication delays.

Scheduling with precedences. Scheduling with precedences to minimize makespan ($\mathsf{P}|\mathsf{prec}|C_{\mathsf{max}}$) is a classic combinatorial optimization problem and is a special case of the communication delay problem with $c = 0$ for all pairs of jobs. In one of the earliest results in the scheduling theory, Graham's list scheduling algorithm [Gra66] was shown to be a 2-factor approximation for the problem. Recently, Svensson [Sve10] gave a matching hardness of approximation result assuming (a variant of) the Unique Games Conjecture [BK09]. When the number of machines is a constant, a series of recent works have obtained $(1+\epsilon)$-approximation in nearly quasi-polynomial time [LR16; Gar18; Kul+20; Li21].

Uniform communication delay setting. The problem becomes much harder with communication delays, even when all the communication delays are equal. This problem is denoted by $\mathsf{P} \mid \mathsf{prec}, c \mid C_{\mathsf{max}}$ and is referred to as scheduling with *uniform* communication delays. In this setting, Graham's list scheduling algorithm obtains a $(c+1)$-factor approximation. This was improved to $2/3 \cdot (c+1)$ by Giroudeau et al. [Gir+08] in the case when the jobs have unit lengths ($\mathsf{P}\infty \mid \mathsf{prec}, p_j = 1, c \geq 2 \mid C_{\mathsf{max}}$). In recent concurrent and independent works, polylogarithmic-factor approximation algorithms have been obtained for the uniform communication delays problem $\mathsf{P} \mid \mathsf{prec}, c \mid C_{\mathsf{max}}$ by Maiti et al. [Mai+20] and Davies et al. [Dav+20; Dav+21].

On the hardness front, when $c = 1$, Hoogeveen, Lenstra and Veltman [HLV94] showed that the problem $\mathsf{P}\infty \mid \mathsf{prec}, p_j = 1, c = 1 \mid C_{\mathsf{max}}$ is NP-hard to approximate to a factor better than $7/6$. The result has been generalized for $c \geq 2$ to $(1 + 1/(c+4))$-hardness [Gir+08].[2]

Scheduling with non-uniform communication delay. We do not know of any algorithm for the non-uniform communication delays ($\mathsf{P}\infty \mid \mathsf{prec}, c_{jk} \mid C_{\mathsf{max}}$) problem. On the hardness side,

[2]Papadimitriou and Yannakakis [PY90] claim a 2-hardness for $\mathsf{P}\infty \mid \mathsf{prec}, p_j = 1, c \mid C_{\mathsf{max}}$, but give no proof. Schuurman and Woeginger [SW99b] remark that "it would be nice to have a proof for this claim".

the best hardness known is the above small constant hardness of the uniform communication delay setting. While our main result shows logarithmic hardness for this problem, it is conceivable that it admits a polylog-approximation algorithm, although our conjectures suggest otherwise.

Duplication model. Scheduling with communication delays problem has also been studied in the duplication model, where we allow jobs to be duplicated (replicated), i.e., executed on more than one machine to avoid communication delays. In this easier model, for the general $\mathsf{P}\infty \mid \mathsf{prec}, p_j, c_{jk}, \mathsf{dup} \mid C_{\mathsf{max}}$ problem, there is a simple 2-factor approximation algorithm by Papadimitriou and Yannakakis [PY90]. On the other hand, [PY90] also show the NP-hardness of $\mathsf{P}\infty \mid \mathsf{prec}, p_j = 1, c, \mathsf{dup} \mid C_{\mathsf{max}}$. Note that the $O(1)$-approximation algorithm for the version with duplication is in sharp contrast to our hardness result (Theorem 183) illustrating that the problem is significantly harder without duplication.

10.1.3 Discussion and Open Problems

While we make progress on the hardness of approximation of scheduling with non-uniform communication delay, the main conceptual contribution of this work is initiating the formal study of the UMPS problem. When jobs have lengths, the problem does not admit a polylogarithmic approximation. However, much less is known for the unit-length case. We now mention a few open problems in this direction.

1. The key open problem is to prove (or disprove) Conjecture 185. A positive resolution of the conjecture would prove the hardness of scheduling related machines with precedences, a long-standing open problem in scheduling theory. By the same reduction as in the proof of Theorem 183, Conjecture 185 also implies a logarithmic hardness of approximation for the non-uniform communication delay problem even when the jobs have unit lengths ($\mathsf{P}\infty \mid \mathsf{prec}, p_j = 1, c_{jk} \mid C_{\mathsf{max}}$).
2. On the other hand, obtaining good approximation algorithms for the UMPS problem would be even more exciting. Is Conjecture 184 true, or is there a polylog-factor approximation algorithm for the unit-length case?

10.1.4 Organization

The rest of the chapter is organized as follows. We first formally define the UMPS problem and relate it to the jobshops problem in Section 10.2. We then use the hardness of the UMPS problem to prove Theorem 183 in Section 10.3. Finally, in Section 10.4, we show that Conjecture 185 implies an improved hardness of related machine scheduling with precedences and that the hypothesis of [BN15] implies a superconstant hardness of the UMPS problem with unit lengths.

10.2 Unique Machine Precedence Constraints Scheduling problem

We first formally define the Unique Machine Precedence constraint Scheduling (UMPS) problem.

Definition 187. *(Unique Machine Precedence constraint Scheduling) In the Unique Machine Precedence constraint Scheduling (UMPS) problem, the input is a set of m machines and n jobs $j_1, j_2, \ldots, j_n$ with precedence relations between them. Furthermore, each job j_l can be scheduled only on a fixed machine $M(l) \in [m]$, and takes $p(l)$ time to complete. The jobs should be scheduled non-preemptively, i.e., once a machine starts processing a job j_l, it has to finish it before processing other jobs. The objective is to schedule the jobs on the corresponding machines in this non-preemptive manner while respecting the precedence relations, so as to minimize the makespan.*

We note that the UMPS problem is a generalization of the classical jobshops problem that we formally define below.

Definition 188. *(Job shops) In the jobshops problem, the input is a set of n jobs to be processed on a set M of m machines. Each job j consists of μ_j operations $O_{1,j}, O_{2,j}, \ldots, O_{\mu_j,j}$. Operation $O_{i,j}$ must be processed for $p_{i,j}$ units of time without interruptions on the machine $m_{i,j} \in M$, and can only be scheduled if all the preceding operations $O_{i',j}, i' < i$ have finished processing. The objective is to schedule all the operations on the corresponding machines to minimize the makespan.*

Note that jobshops problem is a special case of the UMPS problem, corresponding to the case when the precedence DAG is a disjoint union of chains. The jobshops problem has received a lot of attention and played an important role in the development of key algorithmic techniques [Law+93; LMR94]. On the hardness front, Mastrolilli and Svensson showed almost optimal hardness results for the problem in a breakthrough result [MS11].

Theorem 189. *For every constant $\epsilon > 0$, assuming $\mathsf{NP} \not\subseteq \mathsf{ZTIME}\left(n^{(\log n)^{O(1)}}\right)$, there is no polynomial-time $(\log n)^{1-\epsilon}$-factor approximation algorithm for the jobshops problem, where n is the total number of operations in the given jobshops instance.*

As a corollary, we obtain the following hardness result.

Corollary 190. *For every constant $\epsilon > 0$, assuming $\mathsf{NP} \not\subseteq \mathsf{ZTIME}\left(n^{(\log n)^{O(1)}}\right)$, there is no polynomial-time $(\log n)^{1-\epsilon}$-factor approximation algorithm for the UMPS problem.*

10.3 Hardness of Scheduling With Non-Uniform Communication Delays

We now give a reduction from the UMPS problem to the non-uniform communication delay problem, thereby proving the hardness of the non-uniform communication delay problem. We restate the theorem for convenience.

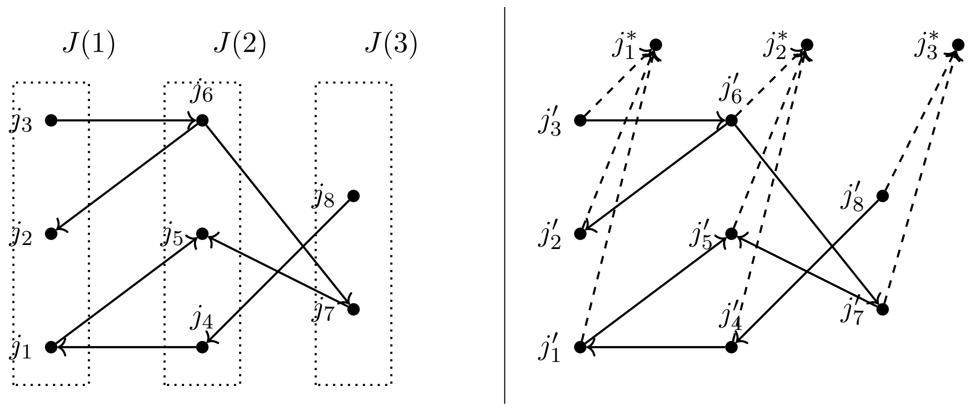

Figure 10.2: Illustration of the reduction from UMPS to non-uniform communication delays. In the communication delay instance on the right, the dashed arrow precedences have communication delay C_∞ while the normal arrow precedences have communication delay 0.

Theorem 183. *For every constant $\epsilon > 0$, assuming $\mathsf{NP} \not\subseteq \mathsf{ZTIME}\left(n^{(\log n)^{O(1)}}\right)$, the non-uniform communication delay problem $(\mathsf{P}\infty \mid \mathsf{prec}, c_{jk} \mid C_{\mathsf{max}})$ does not admit a polynomial-time $(\log n)^{1-\epsilon}$-approximation algorithm.*

Reduction. Let I be an instance of the UMPS problem with n jobs $j_1, j_2, \ldots, j_n$, and m machines. Furthermore, each job j_l has a processing time $p(l)$ and can be scheduled only on the machine $M(l) \in [m]$. For an index $i \in [m]$, let $J(i) \subseteq \{j_1, j_2, \ldots, j_n\}$ denote the set of jobs that can be scheduled on the machine i.

Roughly speaking, our idea in the reduction is to output a non-uniform communication delay instance where we force the jobs in $J(i)$ to be scheduled on the same machine, for every $i \in [m]$. We achieve this by adding a set of m dummy jobs $j_1^*, j_2^*, \ldots, j_m^*$ and adding precedences with very large communication delay from all the jobs in $J(i)$ to j_i^* for every $i \in [m]$. More formally, we define an instance I' of the non-uniform communication delay problem as follows. First, we choose a large integer $C_\infty = n \sum_{l=1}^{n} p(l)$. There are $n+m$ jobs in I': a set of n jobs $j'_1, j'_2, \ldots, j'_n$ such that for each $l \in [n]$, the processing time of j'_l is equal to $p(l)$, and a set $\{j_1^*, j_2^*, \ldots, j_m^*\}$ of m dummy jobs, each with processing time 1. For every precedence relation $j_u \prec j_v$ in the original instance I, there is a precedence relation $j'_u \prec j'_v$ in I' with communication delay 0. Finally, for every i, and every job $j_l \in J(i)$, there is a precedence relation $j'_l \prec j_i^*$ with communication delay C_∞.

Completeness. Suppose that there is a schedule for I with makespan at most L. Then, we claim that there is a schedule for I' with makespan at most $L + 1$. We use m machines and schedule the job j'_l on the machine $M(l)$ in the same time slot used by the schedule for I. As the communication delay of the precedences among the jobs $\{j'_1, j'_2, \ldots, j'_n\}$ is zero, we can schedule these jobs using m machines with makespan at most L. Now, after all the jobs $\{j'_1, j'_2, \ldots, j'_n\}$ have been scheduled, we schedule the job j_i^* in the machine i, for every $i \in [m]$. As we are scheduling all the jobs in $J(i)$ and j_i^* on the same machine for every $i \in [m]$, we incur no communication delay when we are scheduling the dummy jobs, and we can schedule all the dummy jobs $j_1^*, j_2^*, \ldots, j_m^*$ simultaneously in the time slot between L and $L+1$.

Soundness. Suppose that there is a schedule for I' with makespan at most L. Then, we claim that there is a schedule for I with makespan at most L as well.

Note that there is a trivial schedule for I where we schedule each job one by one after topologically sorting them, that has a makespan of $\sum_{j=1}^{n} p(j)$. Thus, henceforth, we assume that $L \leq \sum_{j=1}^{n} p(j)$. For an index $i \in [m]$, let $J'(i)$ be the subset of jobs in I' whose corresponding jobs in I are to be scheduled on the machine i:

$$J'(i) := \{j'_l : M(l) = i\}.$$

We claim that in the schedule for I' with makespan at most L, for every $i \in [m]$, all the jobs in $J'(i)$ must be scheduled on the same machine. Suppose for the sake of contradiction that this is not the case. If there are jobs j'_{l_1} and j'_{l_2} such that $M(l_1) = M(l_2) = i$ are scheduled on different machines i_1, i_2 in the schedule for I', at least one of j'_{l_1} and j'_{l_2} is scheduled on a different machine than j_i^*. However, as there are precedence relations $j'_{l_1} \prec j_i^*$ and $j'_{l_2} \prec j_i^*$ with communication delay C_∞, at least one of the precedence relations has to wait for the communication delay, and thus, the makespan is at least $C_\infty > L$, a contradiction.

Thus, for every $i \in [m]$, all the jobs in $J'(i)$ are processed on the same machine in I'. This implies that at any point of time, at most one job from $J'(i)$ is being processed, for every $i \in [m]$. Using this observation, we output a schedule for I: for every job $j_l \in J(i)$, we schedule j_l in the same time slot used by the job j'_l in the schedule for I'. By the above observation, every machine $i \in [m]$ is used at most once at any time point. Furthermore, as the schedule for I' respects the precedence conditions, the new schedule for I also respects the precedence conditions. Note that the makespan of this schedule for I is equal to L. This completes the proof that there exists a schedule for I with makespan at most L, if there exists a schedule for I' with makespan at most L.

This completes the proof of Theorem 183. We remark that the same reduction also proves a $(\log n)^{1-\epsilon}$-factor inapproximability of the bounded-machines version $\mathsf{P} \mid \mathsf{prec}, c_{jk} \mid C_{\max}$ of the non-uniform communication delay problem.

10.4 Conditional Hardness of Scheduling With Precedence Constraints on Related Machines

In this section, we first prove that Conjecture 185 implies improved hardness of scheduling related machines with precedences.

We begin by formally defining the scheduling related machines with precedences problem ($\mathsf{Q} \mid \mathsf{prec} \mid C_{\max}$).

Definition 191. *(Scheduling related machines with precedences) In the scheduling related machines with precedences problem, the input is a set of m machines $\mathcal{M}$ and a set of n jobs $\mathcal{J}$ with precedences among them. Furthermore, each machine i has speed $s_i \in \mathbb{Z}^+$, and each job j has processing time $p_j \in \mathbb{Z}^+$, and scheduling the job j on machine i takes $\frac{p_j}{s_i}$ units of time. The* *objective is to schedule the jobs on the machines non-preemptively respecting the precedences constraints, to minimize the makespan.*

An algorithm with $O(\log m)$ approximation guarantee for the problem was given independently by Chudak and Shmoys [CS99], and Chekuri and Bender [CB01]. On the hardness side, a hardness factor of 2 follows from the identical machines setting [Sve10], assuming a variant of the Unique Games Conjecture. Furthermore, Bazzi and Norouzi-Fard [BN15] put forth a hypothesis on the hardness of a k-partite graph partitioning problem, which implies a super constant hardness of the scheduling related machines with precedences problem.

We now prove that Conjecture 185 implies poly logarithmic hardness of scheduling related machines with precedences problem.

Theorem 186. *Assuming Conjecture 185 and* $\mathsf{NP} \not\subseteq \mathsf{DTIME}\left(n^{(\log n)^{O(1)}}\right)$*, there exists an absolute constant* $\gamma > 0$ *such that the problem of scheduling related machines with precedences* $(\mathsf{Q} \mid \mathsf{prec} \mid C_{\mathsf{max}})$ *has no polynomial-time* $O\left((\log m)^{\gamma}\right)$*-factor approximation algorithm.*

Reduction. Our reduction is essentially the same reduction as in [BN15] where the authors obtained conditional hardness of the related machine scheduling with precedences problem assuming the hardness of a certain k-partite graph partitioning problem. However, our soundness analysis needs more technical work.

We start with an instance I of the UMPS problem with n unit sized jobs $j_1, j_2, \ldots, j_n$ and m machines, and every job j_l can only be scheduled on the machine $M(l) \in [m]$. Furthermore, we let $J(i) \subseteq [n], i \in [m]$ denote the set of all the jobs that can be scheduled on the machine i.

We now output an instance I' of the related machine scheduling problem. We choose a parameter $\kappa = 10n^3m$. For every $l \in [n]$, we have a set $\mathcal{J}_l$ of $\kappa^{2(m-M(l))}$ jobs in I'. The processing time of each of these jobs is equal to $\kappa^{M(l)-1}$. For every $i \in [m]$, we have $\mathcal{M}_i$, a set of $\kappa^{2(m-i)}$ machines, each with speed κ^{i-1}. Furthermore, for every precedence constraint $j_u \prec j_v$ in I, we have $j'_{l_1} \prec j'_{l_2}$ for every $j'_{l_1} \in \mathcal{J}_u$ and $j'_{l_2} \in \mathcal{J}_v$.

Completeness. Suppose that there is a scheduling of I with makespan equal to L. Then, we claim that there is a scheduling of I' with makespan at most L as well. Note that all the jobs in $\mathcal{J}_l$ can be scheduled on the machines $\mathcal{M}_{M(l)}$ in unit time. We obtain a scheduling of I' by assigning the jobs $\mathcal{J}_l$ to the machines $\mathcal{M}_{M(l)}$ in the time slot used in I to schedule the job j_l. This scheduling of I' is indeed a valid scheduling, and has a makespan of at most L.

Soundness. We prove the soundness part in the lemma below.

Lemma 192. *Suppose that there is a scheduling of* I' *with makespan* L*. Then, we will show that there is a scheduling of* I *with makespan at most* $2L$*.*

Proof. Note that there is a trivial scheduling of I where we schedule jobs in a topological sort one by one, with makespan equal to n. Thus, henceforth, we assume that $L \leq n$.

Let $\gamma = \frac{1}{10n^2}$. We claim that for every $l \in [n]$, at most $\gamma\kappa^{2(m-M(l))}$ jobs in $\mathcal{J}_l$ are processed by machines that do not belong to $\mathcal{M}_{M(l)}$ in the scheduling I'. The proof of this claim follows from Lemma 1 of [BN15], and we present it here for the sake of completeness. Fix an index $l \in [n]$, and for ease of notation, let $i = M(l)$. First, as each job in $\mathcal{J}_l$ has length κ^{i-1}, and the processing speed of each machine in $\mathcal{M}_j, j < i$ is at most $\kappa^{j-1} \leq \kappa^{i-2}$, no job in $\mathcal{J}_l$ is scheduled on machines in $\mathcal{M}_j$, $j < i$, as the makespan of I' is at most $n < \kappa$. Now, consider an integer $j \in [m]$, $j > i$. There are $\kappa^{2(m-j)}$ machines in $\mathcal{M}_j$, and they have a processing speed of κ^{j-1}. Thus, in time $L \leq n$, they can process at most

$$\frac{n \cdot \kappa^{2(m-j)} \cdot \kappa^{j-1}}{\kappa^{i-1}} \leq \frac{n}{\kappa} \cdot \kappa^{2(m-i)}$$

jobs of $\mathcal{J}_l$. Taking union over all $j > i$, we get that at most

$$\frac{nm}{\kappa} \cdot \kappa^{2(m-i)} \leq \frac{1}{10n^2} \kappa^{2(m-i)}$$

jobs in $\mathcal{J}_l$ are processed by machines outside $\mathcal{M}_i$. In other words, for every job j_l of I, at most γ fraction of the jobs in $\mathcal{J}_l$ are processed by machines outside $\mathcal{M}_{M(l)}$.

Now, consider a scheduling of the jobs in I' where for every $l \in [n]$, we get rid of the jobs in $\mathcal{J}_l$ that are processed by machines outside $\mathcal{M}_{M(l)}$. After removing the jobs processed by other machines, we still have that for every $l \in [n]$, at least $1 - \gamma$ fraction of the jobs in $\mathcal{J}_l$ are processed. Also observe that since we are only deleting some jobs, the makespan of the new scheduling is at most L as well. Recall that processing each job in $\mathcal{J}_l$ takes unit time on the machines in $\mathcal{M}_{M(l)}$.

Using this observation, we obtain a fractional scheduling of I in time L as follows. For every $l \in [n]$ and $t \in [L]$, define the variable $x_{l,t}$ to be the fraction of the jobs of $\mathcal{J}_l$ that are scheduled by the machines $\mathcal{M}_{M(l)}$ in the time slot t. By the above discussion, we get the following properties of this fractional scheduling.

1. Every job $l \in [n]$ is almost fully processed. For every $l \in [n]$, we have

$$\sum_{t=1}^{L} x_{l,t} \geq 1 - \gamma$$

2. Every machine is used only for processing a single unit of job in a time slot.

$$\sum_{l \in J(i)} x_{l,t} \leq 1 \; \forall i \in [m], t \in [L].$$

3. If there is a precedence constraint $j_{l_1} \prec j_{l_2}$ in I, l_2's processing is done only in the time slots after l_1 is fully processed. More formally,

$$x_{l_1,t} > 0 \Rightarrow x_{l_2,t'} = 0 \, \forall t' \leq t$$

We will now show that the fractional scheduling implies that the instance I has an integral scheduling with makespan at most $O(L)$, thereby proving the Lemma. We will prove this in two steps: first, we modify the fractional scheduling to obtain another fractional scheduling with better structure, and then next, we use this to obtain the integral scheduling.

For a job $l \in [n]$, define the starting time t_l^s and the end time t_l^e as the minimum and the maximum times at which l is being processed.

$$t_l^s = \min\{t : x_{l,t} > 0\}\,,\; t_l^e = \max\{t : x_{l,t} > 0\}$$

Note that if we have $j_{l_1} \prec j_{l_2}$, $t^s_{l_2} > t^e_{l_1}$. We now modify the fractional scheduling to ensure that each machine processes the job with the lowest ending time first, from the available set of the jobs. More formally, for a machine $i \in [m]$, consider the pair of jobs $l_1, l_2 \in J(i)$ and time slot $t \in [L]$ satisfying the following conditions.

(C1) The job l_1 has lower ending time: $t^e_{l_1} < t^e_{l_2}$, or $t^e_{l_1} = t^e_{l_2}$ and $l_1 < l_2$.

(C2) The job l_1 can be processed on the time slot t, but the job l_2 is processed instead of finishing the job l_1:

$$t^s_{l_1} \le t < t^e_{l_1} \, , \, x_{l_2,t} > 0$$

If there are jobs l_1, l_2 and time slot t satisfying these conditions, we swap the processing times, and process the job l_1 in the time slot t instead of l_2. More formally, let $t' > t$ be such that $x_{l_1,t'} > 0$. Let $y = \min(x_{l_1,t'}, x_{l_2,t})$. We obtain a new fractional scheduling by setting

$$x_{l_1,t'} = x_{l_1,t'} - y \, , \, x_{l_1,t} = x_{l_1,t} + y$$
$$x_{l_2,t'} = x_{l_2,t'} + y \, , \, x_{l_2,t} = x_{l_2,t} - y$$

Note that the operation does not increase the ending time of either job and does not decrease the starting time of either job and thus, results in a valid fractional scheduling respecting the precedence conditions. We repeat the swapping operations until there is no triple i, j, t left where both (C1) and (C2) are true. We also update the starting and ending times of the jobs t^s_l and t^e_l appropriately when we apply the swapping operations.

Next, we apply another transformation to the fractional scheduling by filling the empty slots in the machines, if there are any. More formally, consider a time slot $t \in [L]$ and job $l \in [n]$ such that the following hold.

(D1) The time slot t is not fully utilized:

$$\sum_{l' \in J(M(l))} x_{l',t} < 1$$

(D2) The job l can be scheduled on the time slot t instead of leaving the machine idle: $t^s_l \le t < t^e_l$.

If there is a time slot t and job l such that the above two conditions hold, we fill the empty slot in the time slot t by processing the job l. Let $t' > t$ be such that $x_{l,t'} > 0$. Let $y = \min(x_{l,t'}, 1 - \sum_{l' \in J(M(l))} x_{l',t})$. We set

$$x_{l,t} = x_{l,t} + y \, , \, x_{l,t'} = x_{l,t'} - y$$

We repeat these operations iteratively until no empty slots can be filled. Similar to the previous case, we update the starting and ending times of the jobs appropriately.

After the two types of operations, we obtain a fractional scheduling with the following property: at every time slot t, for a machine $i \in [m]$, let $S_{i,t}$ be the set of jobs that can be scheduled on i in the time slot t:

$$S_{i,t} := \{l \in J(i) : t^s_l \le t \le t^e_l\}$$

We sort the jobs in $S_{i,t}$ as $\{l_1, l_2, \ldots, l_k\}$ by increasing order of ending times, and breaking ties based on the index. The fractional scheduling greedily schedules the jobs $l_1, l_2, \ldots,$ in that order. More formally, we have

$$x_{l_1,t} = \sum_{t'=1}^{L} x_{l_1,t'} - \sum_{t'=1}^{t-1} x_{l_1,t'}$$

and

$$x_{l_2,t} = \min\left(1 - x_{l_1,t}, \sum_{t'=1}^{L} x_{l_2,t'} - \sum_{t'=1}^{t-1} x_{l_2,t'}\right)$$

and so on.

Our goal is to show that in this final fractional scheduling that we obtained, each machine schedules at most two jobs in any time slot. In order to prove this, we first define the following parameter, $P_{i,t}$, the amount of jobs *partially* completed in the machine i by the time t.

$$P_{i,t} = \sum_{l \in J(i): t_l^e > t} \sum_{t'=1}^{t} x_{l,t'}$$

We claim that for every $i \in [m], t \in [L]$, we have $P_{i,t} \leq \gamma t$. Fix a machine $i \in [m]$. We will prove the claim by induction on t.

1. Base case when $t = 1$. If no job is processed by the machine i in the time slot $t = 1$, the claim is trivially satisfied. Else, let l_1 be the job in $J(i)$ with the lowest ending time, breaking ties by the lowest index. Note that the fractional scheduling fully schedules the job l_1 in the time slot $t = 1$. As each job is processed for at least $1 - \gamma$ duration, we get that $P_{i,1}$ is at most γ.
2. Inductive proof. Suppose that the claim holds for all $t' \leq t$ and consider the time slot $t + 1$. For ease of notation, let $S = S_{i,t+1}$ be the set of jobs that can be processed on the machine i in the time slot $t + 1$. If S is empty, the inductive claim trivially holds. Else, let $l \in S$ be the job with the lowest ending time (breaking ties by the least index). Note that the modified fractional scheduling finishes the job l in the time slot $t + 1$. Let $x'_{l,t}$ denote the amount of the job l that is processed by time t i.e., $x'_{l,t} = \sum_{t'=1}^{t} x_{l,t}$. The amount of jobs that are partially finished at the end of time slot $t + 1$ is at most

$$\begin{aligned} P_{i,t+1} &\leq P_{i,t} - x'_{l,t} + (1 - x_{l,t+1}) \\ &\leq P_{i,t} + \gamma \leq (t+1)\gamma \end{aligned}$$

We will now show that every machine processes at most 2 jobs in a time slot. Consider a machine $i \in [m]$ and time slot $t \in [L]$. Let $S_{i,t} := \{l_1, l_2, \ldots, l_k\}$. By the previous claim, we know that at most $(t-1)\gamma$ portion of the job l_u is finished before time t, for every $u \in [k]$. Note that $(t-1)\gamma \leq L\gamma \leq \frac{1}{10n}$. Thus, the greedy fractional scheduling can only schedule at most two jobs, as each of them takes at least $1 - \frac{1}{10n}$ time. Finally, using this observation, we can duplicate every time slot to obtain an integral scheduling of I with makespan at most $2L$. □

Parameter analysis. The number of machines in the related machines scheduling instance is $M = \kappa^{O(m)} = n^{O(m)}$, while the hardness gap is $(\log n)^{1-\epsilon'}$ for every $\epsilon' > 0$. By setting ϵ' appropriately, we get a hardness of $(\log M)^{\Omega(1)}$ for the scheduling related machines with precedences problem.

10.4.1 Hypothesis of [BN15]mplies superconstant hardness of the UMPS problem with unit lengths

Bazzi and Norouzi-Fard [BN15] introduced the following hypothesis and proved that it implies a superconstant hardness for scheduling related machines with precedences.

Hypothesis 193 ([BN15]). *For every $\epsilon, \delta > 0$ and constant integers $k, Q > 0$, the following problem is NP-hard. Given a k-partite graph $G = (V_1, V_2, \ldots, V_k, E_1, E_2, \ldots, E_{k-1})$ with $|V_i| = n$ for all $1 \le i \le k$, and E_i being the set of edges between V_i and V_{i+1} for every $1 \le i < k$, distinguish between the two cases:*

1. *(YES case) Every V_i can be partitioned into $V_{i,0}, V_{i,1}, \ldots, V_{i,Q-1}$ such that*
   - *There is no edge between V_{i,j_1} and V_{i+1,j_2} for all $1 \le i < k, j_1 > j_2 \in [Q]$.*
   - *$|V_{i,j}| \ge \frac{(1-\epsilon)}{Q} n$ for all $i \in [k], j \in [Q]$.*
2. *(NO case) For every $1 < i \le k$, and any two sets S, T with $S \subseteq V_i$, $T \subseteq V_{i-1}$, $|S| = |T| = \delta n$, there is an edge between S and T.*

We now prove that the above hypothesis implies that it is NP-hard to obtain a constant factor approximation algorithm for the UMPS problem, even when all the jobs have unit length.

Reduction. Given an instance of k-partite problem I, we output an instance I' of the UMPS problem as follows: there are $n' = nk$ unit sized jobs in I', one job corresponding to each vertex of G. There are k machines, and all the jobs in $V_i, i \in [k]$ can only be scheduled on the machine i. For every edge $e = (u, v)$ in the graph such that $u \in V_i, v \in V_{i+1}$, we have a precedence condition $u \prec v$ in I'. We choose the parameter $Q = k$, and $\delta = \epsilon = \frac{1}{k}$.

Completeness. Suppose that the YES case of Hypothesis 193 holds i.e., there is a partition of V_i into $V_{i,0}, V_{i,1}, \ldots, V_{i,Q-1}$ respecting the two conditions above. Then, we claim that there is a scheduling of I' with makespan at most $3n$. For every machine $i \in [k]$, we schedule the jobs in $V_{i,0}$ (in arbitrary order), and then the jobs in $V_{i,1}$ (in arbitrary order) and so on. However, we start the execution of the jobs in $V_{i,0}$ at time t_i, and then, execute the jobs in $V_{i,l}$ immediately after the execution of the jobs in $V_{i,l-1}$ for all $l \ge 1$. The parameters t_i, $i \in [k]$ are chosen such that for every pair of jobs u, v with $u \prec v$, u is guaranteed to have scheduled before v. In particular, we choose $t_i = (i-1)n\left(\epsilon + \frac{1}{Q}\right)$.

We now prove that this results in a valid scheduling that respects the precedence conditions. Consider a pair of jobs u, v such that $u \in V_i, v \in V_{i+1}$ such that $u \prec v$. As the k-partite graph satisfies the YES condition, we have integers j_1, j_2 such that $u \in V_{i,j_1}$ and $v \in V_{i+1,j_2}$, and $j_1 \le j_2$. Note that u is processed by time at most

$$t_u = t_i + |V_{i,0}| + |V_{i,1}| + \ldots + |V_{i,j_1}|$$

Furthermore, v is processed only after time

$$t_v = t_{i+1} + |V_{i+1,0}| + |V_{i+1,1}| + \ldots + |V_{i+1,j_2-1}|$$

We have

$$\begin{aligned}
t_v - t_u &= t_{i+1} + |V_{i+1,0}| + |V_{i+1,1}| + \ldots + |V_{i+1,j_2-1}| - (t_i + |V_{i,0}| + |V_{i,1}| + \ldots + |V_{i,j_1}|) \\
&= n\left(\epsilon + \frac{1}{Q}\right) + |V_{i+1,0}| + |V_{i+1,1}| + \ldots + |V_{i+1,j_2-1}| - (n - |V_{i,j_1+1}| + |V_{i,j_1+2}| + \ldots + |V_{i,Q-1}|) \\
&\geq n\left(\epsilon + \frac{1}{Q}\right) + j_2\frac{(1-\epsilon)n}{Q} - \left(n - \frac{(Q-j_1-1)(1-\epsilon)n}{Q}\right) \\
&\geq n\left(\epsilon + \frac{1}{Q}\right) + j_1\frac{(1-\epsilon)n}{Q} - \left(j_1\frac{(1-\epsilon)n}{Q} + \epsilon n + \frac{(1-\epsilon)n}{Q}\right) \geq 0
\end{aligned}$$

Thus, the schedule is a valid scheduling of I'. The makespan of this scheduling is at most $t_k + n = (k-1)n\left(\epsilon + \frac{1}{Q}\right) + n \leq 3n$.

Soundness. Suppose that the NO case of Hypothesis 193 holds. We claim that in this case, the makespan of I' is at least $(1-2\delta)kn$. For every $i \in [k]$, let s_i denote the time at which the machine i has finished $(1-\delta)n$ jobs of V_i. For an index $i \in [k]$, let $S(i) \subseteq V_i$ denote the set of jobs that are not processed by the time s_i. By the definition of s_i, we have $|S(i)| \geq \delta n$. By the NO case of Hypothesis 193, we get that there are at least $(1-\delta)n$ jobs in V_{i+1} that have dependencies in $S(i)$. Note that all these jobs can be scheduled only after s_i. Thus, we get

$$s_{i+1} \geq s_i + (1-2\delta)n \ \forall i \in [k-1]$$

Summing over all i, we get that the makespan of the scheduling is at least $(1-2\delta)kn$, which is at least $\frac{kn}{2}$ when $k \geq 4$. By choosing k large enough, this completes the proof that assuming Hypothesis 193, it is NP-hard to obtain a $O(1)$ factor approximation algorithm for the UMPS problem when the jobs have unit lengths.

Chapter 11

Conclusion

We conclude by mentioning a few directions for further research.

Boolean PCSP Dichotomy. Can we show that every Boolean PCSP is either in P or is NP-Hard?

Problem 194. *When can a Boolean PCSP Γ be solved in polynomial time? Is it true that every Boolean PCSP can be solved in polynomial time or is NP-Hard?*

For the case of CSPs (on general domains), a CSP has a polynomial time algorithm if and only if it has a cyclic[1] polymorphism of arity at least 3. The hardness part was proved in the early 2000s itself [BJK05] and proving the algorithmic part was the main challenge [Bul17; Zhu20]. However, for Promise CSPs, even in the Boolean case, we do not have a candidate characterization of polymorphisms that leads to polynomial time algorithms.

On the algorithmic side, the best result is that symmetric polymorphisms of arbitrarily large arity lead to algorithms [Bra+20] using a combination of the basic LP relaxation and the affine relaxation. Could it be true that using the basic SDP relaxation together with the affine relaxation gives an optimal algorithm for all Boolean PCSPs? We shed some light on this question in Chapter 6 where we study the power of the basic SDP relaxation for PCSPs. But our result applies to the basic SDP alone, and a potential avenue to understand the power of basic SDP together with the Affine relaxation to study the minion $\mathcal{M}_{\mathrm{SDP+Aff}}$ such that a PCSP Γ can be solved by the basic SDP relaxation together with the affine relaxation if and only if there is a minion homomorphism[2] from $\mathcal{M}_{\mathrm{SDP+Aff}}$ to $\mathrm{Pol}(\Gamma)$.

On the hardness side, as seen in Chapter 3, the rich 2-to-1 conjecture [BKM21] could be of help in obtaining improved NP-Hardness results. In Chapter 3, we showed that when the Boolean PCSP contains the predicate $x \leq y$, i.e., when the polymorphisms are all monotone functions, if every polymorphism contains a coordinate of high Shapley influence, then the underlying PCSP is NP-Hard (under the rich 2-to-1 conjecture). Can we extend this result to arbitrary Boolean functions using a suitable generalization of Shapley influence?

NP-Hardness of Approximate Graph Coloring. Despite great progress on Promise CSPs,

[1]A function f of arity $l \geq 2$ is said to be cyclic if $f(x_1, x_2, \dots, x_l) = f(x_2, x_3, \dots, x_l, x_1) = \dots = f(x_l, x_1, \dots, x_{l-1})$ for every $x_1, x_2, \dots, x_l$.

[2]A Minion homomorphism $f : \mathcal{M}_1 \to \mathcal{M}_2$ (formally defined in Chapter 6) is a mapping that preserves the arity of the elements and also commutes with taking minors.

we still do not know if it is NP-Hard to 6-color a 3-colorable graph in polynomial time. More generally, let (c, s)-approximate graph coloring be the computational problem of coloring a graph that is promised to be c-colorable with s colors.

Problem 195. *Prove that $(3, s)$-approximate graph coloring is NP-Hard for every constant $s \geq 3$.*

In Chapter 4, we have proved that d-to-1 conjecture [Kho02b] for any constant d implies that it is NP-Hard to color a 3-colorable graphs with $O(1)$ colors, thus resolving Problem 195. The imperfect completeness version of d-to-1 conjecture when $d = 2$ was proved recently in a breakthrough series of works [KMS17; Din+18b; Din+18a; KMS18]. However, their result starts with the hardness of linear equations as a starting point, and thus, does not extend to the case when the problem has perfect completeness. Can we obtain a proof of d-to-1 (perhaps with larger d) conjecture by using a different problem as a starting point, and studying analogous objects to Grassman graphs of [KMS18]?

A different approach to Problem 195 is using the PCSP machinery developed recently. Barto, Bulin, Krokhin, Opršal [Bar+21] proved that 5-coloring a 3-colorable graph is NP-Hard. They achieve this by showing that there is a minion homomorphism from the polymorphisms of this PCSP to the polymorphisms of $O(1)$-coloring a 2-colorable 3-uniform hypergraphs. By the result of Dinur, Regev, Smyth [DRS05], it is NP-Hard to color a 2-colorable 3-uniform hypergraph with $O(1)$ colors, thus implying the hardness of $(3, 5)$-approximate graph coloring. Approximate graph coloring is a special case of a more general graph homomorphism problem where the input is a pair of graphs H_1, H_2 such that there is a homomorphism $H_1 \to H_2$, given an input graph G with the promise that $G \to H_1$, can we find a homomorphism $G \to H_2$? (c, s)-approximate graph coloring is the case when $H_1 = K_c, H_2 = K_s$. Recently, Krokhin, Opršal, Wrochna, Zivny [Kro+20] proved the NP-Hardness of the graph homomorphism problem when $H_1 = C_l, H_2 = K_2$ for every odd integer $l \geq 3$. A possible avenue towards proving Problem 195 is to generalize their techniques for the case when H_2 is a larger clique.

Robust algorithms for PCSPs. In Chapter 6, we initiated the study of robust algorithms for PCSPs. Characterizing which PCSPs is interesting on its own, but more so in connection with the power of SDP algorithms for PCSPs (Conjecture 65).

The main goal is to answer the following question:

Problem 196. *Which Promise CSPs have Robust Algorithms? Is it the same class as Promise CSPs that can be solved by the basic SDP relaxation?*

As we proved in Chapter 6, the existence of robust algorithms for a PCSP is characterized by its polymorphisms. We have shown that having AT or MAJ polymorphisms of all odd arities leads to algorithms. On the other hand, the CSP 3-LIN, which has Parity polymorphisms of all odd arities, does not admit a robust algorithm. This begs the question: which polymorphism families lead to robust algorithms? Intuitively, the polymorphisms should be robust to noise, and the challenge is to precisely characterize the notion of noise stability that leads to algorithms and incorporate it into the robust algorithms.

On the hardness side, proving integrality gaps for the basic SDP relaxation and using Raghavendra's [Rag08] framework is a general technique to show robust hardness. Using Lemma 86, we can reduce the problem of finding integrality gaps for the basic SDP of a PCSP to finding sphere colorings $f : \mathbb{S}^n \to [r]$ satisfying certain structural properties. We have used results from sphere Ramsey theory to answer such questions, leading to robust hardness for some classes of PCSPs, but the general problem of finding specific structures in sphere colorings is wide open.

Inapproximability of Related Machines Scheduling with Precedences. In the related machine scheduling, there are n jobs with processing times $p_1, p_2, \ldots, p_n$ with precedence constraints between them. That is, we are given a DAG over these jobs and if there is an edge from job j_1 to j_2, the processing of j_2 can only begin after j_1 finishes. There are m machines, each with speed $s_i, i \in [m]$. A job j on a machine i takes time $\frac{p_j}{s_i}$. The objective is to schedule the jobs on the machines to minimize the makespan. The problem admits a $\log m$ factor approximation algorithm [CB01; CS99]. On the hardness side, the best known inapproximability is a factor 2 [Sve10; BK09], assuming a variant of the Unique Games Conjecture. Whether we can get a $O(1)$ approximation algorithm for the problem remained a long-standing open problem in scheduling theory, and has been asked in multiple influential surveys [SW99a; Ban17; WS11].

Problem 197. *Is there an $O(1)$ factor approximation algorithm for related machine scheduling with precedences problem?*

As we proved in Chapter 10, improved hardness of the Unique Machine Precedence Scheduling (UMPS) problem Conjecture 185 implies poly logarithmic hardness of scheduling related machines with precedences problem. A potential avenue to showing the hardness of the UMPS problem is via using the rich 2-to-1 conjecture [BKM21] in the framework of Bansal and Khot [BK09].

Bibliography

[ABP19] Per Austrin, Amey Bhangale, and Aditya Potukuchi. "Simplified inpproximability of hypergraph coloring via t-agreeing families". In: *CoRR* abs/1904.01163 (2019). arXiv: `1904.01163`.

[ABP20] Per Austrin, Amey Bhangale, and Aditya Potukuchi. "Improved Inapproximability of Rainbow Coloring". In: *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020*. 2020, pp. 1479–1495.

[AD22] Albert Atserias and Víctor Dalmau. "Promise Constraint Satisfaction and Width". In: *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM. 2022, pp. 1129–1153.

[AGH17] Per Austrin, Venkatesan Guruswami, and Johan Håstad. "(2+ϵ)-SAT Is NP-hard". In: *SIAM J. Comput.* 46.5 (2017), pp. 1554–1573.

[Aha01] Ron Aharoni. "Ryser's Conjecture for Tripartite 3-Graphs". In: *Comb.* 21.1 (2001), pp. 1–4.

[AHK96] Ron Aharoni, Ron Holzman, and Michael Krivelevich. "On a Theorem of Lovász on Covers in tau-Partite Hypergraphs". In: *Combinatorica* 16.2 (1996), pp. 149–174.

[AKS11] Per Austrin, Subhash Khot, and Muli Safra. "Inapproximability of Vertex Cover and Independent Set in Bounded Degree Graphs". In: *Theory Comput.* 7.1 (2011), pp. 27–43.

[Alo+98] Noga Alon, Yossi Azar, János Csirik, Leah Epstein, Sergey V. Sevastianov, Arjen P. A. Vestjens, and Gerhard J. Woeginger. "On-Line and Off-Line Approximation Algorithms for Vector Covering Problems". In: *Algorithmica* 21.1 (1998), pp. 104–118.

[APV16] Andris Ambainis, Krišjānis Prūsis, and Jevgēnijs Vihrovs. "Sensitivity Versus Certificate Complexity of Boolean Functions". In: *Proceedings of the 11th International Computer Science Symposium on Computer Science — Theory and Applications - Volume 9691*. CSR 2016. St. Petersburg, Russia, 2016, pp. 16–28.

[Aro+98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. "Proof Verification and the Hardness of Approximation Problems". In: *J. ACM* 45.3 (1998), pp. 501–555.

[AS98] Sanjeev Arora and Shmuel Safra. "Probabilistic Checking of Proofs: A New Characterization of NP". In: *Journal of the ACM* 45.1 (1998), pp. 70–122.

[AZ20] Ron Aharoni and Shira Zerbib. "A generalization of Tuza's conjecture". In: *Journal of Graph Theory* 94.3 (2020), pp. 445–462.

[Aza+13] Yossi Azar, Ilan Reuven Cohen, Seny Kamara, and F. Bruce Shepherd. "Tight bounds for online vector bin packing". In: *Symposium on Theory of Computing Conference, STOC'13*. 2013, pp. 961–970.

[Ban17] Nikhil Bansal. "Scheduling Open Problems: Old And New." In: *MAPSP 2017* www.mapsp2017.ma.tum.de/MAPSP2017-Bansal.pdf (2017).

[Bar+21] Libor Barto, Jakub Bulín, Andrei A. Krokhin, and Jakub Opršal. "Algebraic Approach to Promise Constraint Satisfaction". In: *J. ACM* 68.4 (2021), 28:1–28:66.

[Bar18a] Libor Barto. Personal communication. 2018.

[Bar18b] Libor Barto. "Cyclic operations in promise constraint satisfaction problems". In: *Dagstuhl workshop The Constraint Satisfaction Problem: Complexity and Approximability, Schloss Dagstuhl, Germany* (2018). Available at `https://www2.karlin.mff.cuni.cz/~barto/Articles/Barto_Dagstuhl18.pdf`.

[BCD20] Patrick Bennett, Ryan Cushman, and Andrzej Dudek. *Closing the Random Graph Gap in Tuza's Conjecture Through the Online Triangle Packing Process*. 2020. arXiv: `2007.04478 [math.CO]`.

[BCS09] Nikhil Bansal, Alberto Caprara, and Maxim Sviridenko. "A New Approximation Method for Set Covering Problems, with Applications to Multidimensional Bin Packing". In: *SIAM J. Comput.* 39.4 (2009), pp. 1256–1278.

[BE51] NG de Bruijn and P Erdos. "A colour problem for infinite graphs and a problem in the theory of relations". In: *Indigationes Mathematicae* 13 (1951), pp. 371–373.

[BEK16] Nikhil Bansal, Marek Eliáš, and Arindam Khan. "Improved Approximation for Vector Bin Packing". In: *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016*. 2016, pp. 1561–1579.

[BFG19] Fábio Botler, Cristina G. Fernandes, and Juan Gutiérrez. "On Tuza's Conjecture for Triangulations and Graphs with Small Treewidth". In: *Proceedings of the tenth Latin and American Algorithms, Graphs and Optimization Symposium, LAGOS 2019, Belo Horizonte, Brazil, June 2-7, 2019*. Vol. 346. Electronic Notes in Theoretical Computer Science. Elsevier, 2019, pp. 171–183.

[BFM10] Tom Bohman, Alan M. Frieze, and Dhruv Mubayi. "Coloring H-free hypergraphs". In: *Random Struct. Algorithms* 36.1 (2010), pp. 11–25.

[BG16] Joshua Brakensiek and Venkatesan Guruswami. "New Hardness Results for Graph and Hypergraph Colorings". In: *31st Conference on Computational Complexity, CCC 2016*. 2016, 14:1–14:27.

[BG17] Joshua Brakensiek and Venkatesan Guruswami. "The Quest for Strong Inapproximability Results with Perfect Completeness". In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2017, August 16-18, 2017, Berkeley, CA, USA*. 2017, 4:1–4:20.

[BG21a] Joshua Brakensiek and Venkatesan Guruswami. “Promise Constraint Satisfaction: Algebraic Structure and a Symmetric Boolean Dichotomy”. In: *SIAM J. Comput.* 50.6 (2021), pp. 1663–1700.

[BG21b] Joshua Brakensiek and Venkatesan Guruswami. “Promise Constraint Satisfaction: Algebraic Structure and a Symmetric Boolean Dichotomy”. In: *SIAM J. Comput.* 50.6 (2021), pp. 1663–1700.

[BGL15] Vijay V. S. P. Bhattiprolu, Venkatesan Guruswami, and Euiwoong Lee. “Approximate Hypergraph Coloring under Low-discrepancy and Related Promises”. In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2015*. 2015, pp. 152–174.

[BGS21] Joshua Brakensiek, Venkatesan Guruswami, and Sai Sandeep. “Conditional Dichotomy of Boolean Ordered Promise CSPs”. In: *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference)*. Vol. 198. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, 37:1–37:15.

[BGS98] Mihir Bellare, Oded Goldreich, and Madhu Sudan. “Free Bits, PCPs, and Nonapproximability-Towards Tight Results”. In: *SIAM J. Comput.* 27.3 (1998), pp. 804–915.

[Bha18] Amey Bhangale. “NP-Hardness of Coloring 2-Colorable Hypergraph with Poly-Logarithmically Many Colors”. In: *45th International Colloquium on Automata, Languages, and Programming*. 2018, 15:1–15:11.

[BJK05] Andrei A. Bulatov, Peter Jeavons, and Andrei A. Krokhin. “Classifying the Complexity of Constraints Using Finite Algebras”. In: *SIAM J. Comput.* 34.3 (2005), pp. 720–742.

[BK09] Nikhil Bansal and Subhash Khot. “Optimal Long Code Test with One Free Bit”. In: *50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009, October 25-27, 2009, Atlanta, Georgia, USA*. IEEE Computer Society, 2009, pp. 453–462.

[BK10] Nikhil Bansal and Subhash Khot. “Inapproximability of Hypergraph Vertex Cover and Applications to Scheduling Problems”. In: *Automata, Languages and Programming, 37th International Colloquium, ICALP 2010, Bordeaux, France, July 6-10, 2010, Proceedings, Part I*. 2010, pp. 250–261.

[BK14a] Nikhil Bansal and Arindam Khan. “Improved Approximation Algorithm for Two-Dimensional Bin Packing”. In: *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014*. 2014, pp. 13–25.

[BK14b] Libor Barto and Marcin Kozik. “Constraint Satisfaction Problems Solvable by Local Consistency Methods”. In: *J. ACM* 61.1 (2014), 3:1–3:19.

[BK16] Libor Barto and Marcin Kozik. “Robustly Solvable Constraint Satisfaction Problems”. In: *SIAM J. Comput.* 45.4 (2016), pp. 1646–1669.

[BKM21] Mark Braverman, Subhash Khot, and Dor Minzer. “On Rich 2-to-1 Games”. In: *12th Innovations in Theoretical Computer Science Conference, ITCS 2021*. Vol. 185. LIPIcs. 2021, 27:1–27:20.

[BKW17] Libor Barto, Andrei Krokhin, and Ross Willard. "Polymorphisms, and How to Use Them". In: *The Constraint Satisfaction Problem: Complexity and Approximability*. Vol. 7. Dagstuhl Follow-Ups. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017, pp. 1–44.

[BN15] Abbas Bazzi and Ashkan Norouzi-Fard. "Towards Tight Lower Bounds for Scheduling Problems". In: *Algorithms - ESA 2015 - 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings*. Vol. 9294. Lecture Notes in Computer Science. Springer, 2015, pp. 118–129.

[Bra+20] Joshua Brakensiek, Venkatesan Guruswami, Marcin Wrochna, and Stanislav Živný. "The Power of the Combined Basic Linear Programming and Affine Relaxation for Promise Constraint Satisfaction Problems". In: *SIAM J. Comput.* 49.6 (2020), pp. 1232–1248.

[Bra+21] Joshua Brakensiek, Neng Huang, Aaron Potechin, and Uri Zwick. "On the mysteries of MAX NAE-SAT". In: *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM. 2021, pp. 484–503.

[BS04] Nikhil Bansal and Maxim Sviridenko. "New approximability and inapproximability results for 2-dimensional Bin Packing". In: *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2004*. 2004, pp. 196–203.

[Bul17] Andrei A. Bulatov. "A Dichotomy Theorem for Nonuniform CSPs". In: *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017*. IEEE Computer Society, 2017, pp. 319–330.

[BŽ21] Alex Brandts and Stanislav Živný. "Beyond PCSP(1-in-3, NAE)". In: *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference)*. Vol. 198. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, 121:1–121:14.

[Cae94] D. de Caen. "The current status of Turán's problem on hypergraphs". In: *Extremal Problems for Finite Sets* 1991, Bolyai Soc. Math. Stud., Vol. 3, pp. 187–197, János Bolyai Math. Soc., Budapest (1994).

[Car+12] Jean Cardinal, Marek Karpinski, Richard Schmied, and Claus Viehmann. "Approximating vertex cover in dense hypergraphs". In: *Journal of discrete algorithms* 13 (2012), pp. 67–77.

[CB01] Chandra Chekuri and Michael A. Bender. "An Efficient Approximation Algorithm for Minimizing Makespan on Uniformly Related Machines". In: *J. Algorithms* 41.2 (2001), pp. 212–224.

[CC06] Miroslav Chlebik and Janka Chlebikova. "Inapproximability Results for Orthogonal Rectangle Packing Problems with Rotations". In: *Algorithms and Complexity, 6th Italian Conference, CIAC 2006*. 2006, pp. 199–210.

[CD72] J Csima and B.N Datta. "The DAD theorem for symmetric non-negative matrices". In: *Journal of Combinatorial Theory, Series A* 12.1 (1972), pp. 147–152.

[CFS08] L. Sunil Chandran, Mathew C. Francis, and Naveen Sivadasan. "Boxicity and maximum degree". In: *J. Comb. Theory, Ser. B* 98.2 (2008), pp. 443–445.

[Cha+20] Parinya Chalermsook, Samir Khuller, Pattara Sukprasert, and Sumedha Uniyal. "Multi-transversals for Triangles and the Tuza's Conjecture". In: *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*. Ed. by Shuchi Chawla. SIAM, 2020, pp. 1955–1974.

[Cha16] Siu On Chan. "Approximation Resistance from Pairwise-Independent Subgroups". In: *J. ACM* 63.3 (2016), 27:1–27:32.

[Cho+11] Mosharaf Chowdhury, Matei Zaharia, Justin Ma, Michael I. Jordan, and Ion Stoica. "Managing data transfers in computer clusters with orchestra". In: *Proceedings of the ACM SIGCOMM 2011 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, Toronto, ON, Canada, August 15-19, 2011*. ACM, 2011, pp. 98–109.

[Chr+17] Henrik I. Christensen, Arindam Khan, Sebastian Pokutta, and Prasad Tetali. "Approximation and online algorithms for multidimensional bin packing: A survey". In: *Comput. Sci. Rev.* 24 (2017), pp. 63–79.

[CJK01] János Csirik, David S. Johnson, and Claire Kenyon. "Better approximation algorithms for bin covering". In: *Proceedings of the Twelfth Annual Symposium on Discrete Algorithms, 2001*. 2001, pp. 557–566.

[CK04] Chandra Chekuri and Sanjeev Khanna. "On Multidimensional Packing Problems". In: *SIAM J. Comput.* 33.4 (2004), pp. 837–851.

[CKL20] Vincent Cohen-Addad, Karthik C.S., and Euiwoong Lee. "On Approximability of k-means, k-median, and k-minsum Clustering". In: *Manuscript* (2020).

[CKL21] Vincent Cohen-Addad, Karthik C. S., and Euiwoong Lee. "On Approximability of Clustering Problems Without Candidate Centers". In: *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*. To appear. 2021.

[CL99] Fan Chung and Linyuan Lu. "An upper bound for the Turán number t3(n, 4)". In: *Journal of Combinatorial Theory, Series A* 87.2 (1999), pp. 381–389.

[CMM09] Moses Charikar, Konstantin Makarychev, and Yury Makarychev. "Near-optimal algorithms for maximum constraint satisfaction problems". In: *ACM Trans. Algorithms* 5.3 (2009), 32:1–32:14.

[CS00] Artur Czumaj and Christian Scheideler. "A new algorithm approach to the general Lovász local lemma with applications to scheduling and satisfiability problems (extended abstract)". In: *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*. ACM, 2000, pp. 38–47.

[CS99] Fabián A. Chudak and David B. Shmoys. "Approximation Algorithms for Precedence-Constrained Scheduling Problems on Parallel Machines that Run at Different Speeds". In: *J. Algorithms* 30.2 (1999), pp. 323–343.

[Cyg13] Marek Cygan. "Improved Approximation for 3-Dimensional Matching via Bounded Pathwidth Local Search". In: *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, Berkeley, CA, USA*. 2013.

[CŽ22a] Lorenzo Ciardo and Stanislav Živný. "CLAP: A New Algorithm for Promise CSPs". In: *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM. 2022, pp. 1057–1068.

[CŽ22b] Lorenzo Ciardo and Stanislav Živný. "The Sherali-Adams Hierarchy for Promise CSPs through Tensors". In: *CoRR* abs/2203.02478 (2022). arXiv: `2203.02478`.

[Dav+20] Sami Davies, Janardhan Kulkarni, Thomas Rothvoss, Jakub Tarnawski, and Yihao Zhang. "Scheduling with Communication Delays via LP Hierarchies and Clustering". In: *To appear at 61st Annual IEEE Symposium on Foundations of Computer Science (FOCS 2020), 16-19 November 2006, Durham, North Carolina, USA, Proceedings* (2020).

[Dav+21] Sami Davies, Janardhan Kulkarni, Thomas Rothvoss, Jakub Tarnawski, and Yihao Zhang. "Scheduling with Communication Delays via LP Hierarchies and Clustering II: Weighted Completion Times on Related Machines". In: *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*. Ed. by Dániel Marx. SIAM, 2021, pp. 2958–2977.

[Dav+22] Sami Davies, Janardhan Kulkarni, Thomas Rothvoss, Sai Sandeep, Jakub Tarnawski, and Yihao Zhang. "On the Hardness of Scheduling With Non-Uniform Communication Delays". In: *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM. 2022, pp. 316–328.

[DDS17] Anindya De, Ilias Diakonikolas, and Rocco A. Servedio. "The Inverse Shapley value problem". In: *Games Econ. Behav.* 105 (2017), pp. 122–147.

[Din+05] Irit Dinur, Venkatesan Guruswami, Subhash Khot, and Oded Regev. "A New Multi-layered PCP and the Hardness of Hypergraph Vertex Cover". In: *SIAM J. Comput.* 34.5 (2005), pp. 1129–1146.

[Din+18a] Irit Dinur, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra. "On non-optimally expanding sets in Grassmann graphs". In: *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018,* ACM, 2018, pp. 940–951.

[Din+18b] Irit Dinur, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra. "Towards a proof of the 2-to-1 games conjecture?" In: *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018,* ACM, 2018, pp. 376–389.

[Din07] Irit Dinur. "The PCP theorem by gap amplification". In: *Journal of the ACM* 54.3 (2007), p. 12.

[DLR95] Richard A. Duke, Hanno Lefmann, and Vojtech Rödl. "On Uncrowded Hypergraphs". In: *Random Struct. Algorithms* 6.2/3 (1995), pp. 209–212.

[DMR09] Irit Dinur, Elchanan Mossel, and Oded Regev. "Conditional Hardness for Approximate Coloring". In: *SIAM J. Comput.* 39.3 (2009), pp. 843–873.

[DR06] Irit Dinur and Omer Reingold. "Assignment Testers: Towards a Combinatorial Proof of the PCP Theorem". In: *SIAM J. Comput.* 36.4 (2006), pp. 975–1024.

[DRS05] Irit Dinur, Oded Regev, and Clifford D. Smyth. “The Hardness of 3-Uniform Hypergraph Coloring”. In: *Combinatorica* 25.5 (2005), pp. 519–535.

[DS05] Irit Dinur and Samuel Safra. “On the Hardness of Approximating Minimum Vertex Cover”. In: *Annals of Mathematics* 162.1 (2005), pp. 439–485.

[DS14] Irit Dinur and David Steurer. “Analytical approach to parallel repetition”. In: *Symposium on Theory of Computing, STOC 2014*. 2014, pp. 624–633.

[Erd81] Paul Erdős. “On the combinatorial problems which I would most like to see solved”. In: *Combinatorica* 1.1 (1981), pp. 25–42.

[Erd88] Paul Erdös. “Problems and Results in Combinatorial Analysis and Graph Theory”. In: *Graph Theory and Applications, Proceedings of the First Japan Conference on Graph Theory and Applications* (1988), pp. 81–92.

[Fei98] Uriel Feige. “A Threshold of ln n for Approximating Set Cover”. In: *J. ACM* 45.4 (1998), pp. 634–652.

[Fic+19] Miron Ficak, Marcin Kozik, Miroslav Olsák, and Szymon Stankiewicz. “Dichotomy for Symmetric Boolean PCSPs”. In: *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019*. Vol. 132. LIPIcs. 2019, 57:1–57:12.

[FK15] Alan Frieze and Michał Karoński. *Introduction to Random Graphs*. Cambridge University Press, 2015.

[FK91] H. Furstenberg and Y. Katznelson. “A density version of the Hales-Jewett theorem.” English. In: *J. Anal. Math.* 57 (1991), pp. 64–119. ISSN: 0021-7670; 1565-8538/e.

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

[FM08] Alan M. Frieze and Dhruv Mubayi. “On the Chromatic Number of Simple Triangle-Free Triple Systems”. In: *Electron. J. Comb.* 15.1 (2008).

[FM13] Alan M. Frieze and Dhruv Mubayi. “Coloring simple hypergraphs”. In: *J. Comb. Theory, Ser. B* 103.6 (2013), pp. 767–794.

[Fra+17] Nevena Francetic, Sarada Herke, Brendan D. McKay, and Ian M. Wanless. “On Ryser’s conjecture for linear intersecting multipartite hypergraphs”. In: *Eur. J. Comb.* 61 (2017), pp. 91–105.

[Fre04] Robert M Freund. “Introduction to semidefinite programming (SDP)”. In: *Massachusetts Institute of Technology* (2004), pp. 11–12. URL: `https://ocw.mit.edu/courses/6-251j-introduction-to-mathematical-programming-fall-2009/08bbc2660764c4f61bde5363ae134339_MIT6_251JF09_SDP.pdf`.

[FS02] Uriel Feige and Christian Scheideler. “Improved Bounds for Acyclic Job Shop Scheduling”. In: *Comb.* 22.3 (2002), pp. 361–399.

[FV98] Tomás Feder and Moshe Y. Vardi. “The Computational Structure of Monotone Monadic SNP and Constraint Satisfaction: A Study through Datalog and Group Theory”. In: *SIAM J. Comput.* 28.1 (1998), pp. 57–104.

[Gar+76] M. R. Garey, Ronald L. Graham, David S. Johnson, and Andrew Chi-Chih Yao. “Resource Constrained Scheduling as Generalized Bin Packing”. In: *J. Comb. Theory, Ser. A* 21.3 (1976), pp. 257–298.

[Gar18] Shashwat Garg. “Quasi-PTAS for Scheduling with Precedences using LP Hierarchies”. In: *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*. 2018, 59:1–59:13.

[GCL18] Yuanxiang Gao, Li Chen, and Baochun Li. “Spotlight: Optimizing Device Placement for Training Deep Neural Networks”. In: *Proceedings of the 35th International Conference on Machine Learning*. Vol. 80. Proceedings of Machine Learning Research. Stockholmsmässan, Stockholm Sweden: PMLR, 2018, pp. 1676–1684.

[GHS02] Venkatesan Guruswami, Johan Håstad, and Madhu Sudan. “Hardness of Approximate Hypergraph Coloring”. In: *SIAM J. Comput.* 31.6 (2002), pp. 1663–1686.

[Gir+08] Rodolphe Giroudeau, Jean-Claude König, Farida Kamila Moulai, and Jérôme Palaysi. “Complexity and approximation for precedence constrained scheduling problems with large communication delays”. In: *Theor. Comput. Sci.* 401.1-3 (2008), pp. 107–119.

[GK99] Sudipto Guha and Samir Khuller. “Greedy strikes back: Improved facility location algorithms”. In: *Journal of algorithms* 31.1 (1999), pp. 228–248.

[GL17] Venkatesan Guruswami and Euiwoong Lee. “Inapproximability of H-Transversal/Packing”. In: *SIAM J. Discret. Math.* 31.3 (2017), pp. 1552–1571.

[GL18] Venkatesan Guruswami and Euiwoong Lee. “Strong Inapproximability Results on Balanced Rainbow-Colorable Hypergraphs”. In: *Combinatorica* 38.3 (2018), pp. 547–599.

[GO05] Venkatesan Guruswami and Ryan O’Donnell. *The PCP Theorem and Hardness of Approximation: Notes on Lectures 7–9*. `https://courses.cs.washington.edu/courses/cse533/05au/`. 2005.

[Gol+01] Leslie Ann Goldberg, Mike Paterson, Aravind Srinivasan, and Elizabeth Sweedyk. “Better Approximation Guarantees for Job-Shop Scheduling”. In: *SIAM J. Discret. Math.* 14.1 (2001), pp. 67–92.

[GOS20] Venkatesan Guruswami, Jakub Opršal, and Sai Sandeep. “Revisiting Alphabet Reduction in Dinur’s PCP”. In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2020, August 17-19, 2020, Virtual Conference*. Vol. 176. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020, 34:1–34:14.

[Gra+79] R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan. “Optimization and approximation in deterministic sequencing and scheduling: a survey”. In: *Ann. Discrete Math.* 4 (1979), pp. 287–326.

[Gra66] R. L. Graham. “Bounds for Certain Multiprocessing Anomalies”. In: *Bell System Technical Journal* 45.9 (1966), pp. 1563–1581.

[GS17] Venkatesan Guruswami and Rishi Saket. “Hardness of Rainbow Coloring Hypergraphs”. In: *37th IARCS Annual Conference on Foundations of Software Technology* *and Theoretical Computer Science, FSTTCS 2017, December 11-15, 2017, Kanpur, India*. 2017, 33:33–33:15.

[GS20a] Venkatesan Guruswami and Sai Sandeep. "d-To-1 Hardness of Coloring 3-Colorable Graphs with O(1) Colors". In: *47th International Colloquium on Automata, Languages, and Programming, ICALP 2020, July 8-11, 2020, Saarbrücken, Germany (Virtual Conference)*. Vol. 168. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020, 62:1–62:12.

[GS20b] Venkatesan Guruswami and Sai Sandeep. "Rainbow Coloring Hardness via Low Sensitivity Polymorphisms". In: *SIAM J. Discret. Math.* 34.1 (2020), pp. 520–537.

[GS22] Venkatesan Guruswami and Sai Sandeep. "Approximate Hypergraph Vertex Cover and generalized Tuza's conjecture". In: *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM. 2022, pp. 927–944.

[GSS15] Venkatesan Guruswami, Sushant Sachdeva, and Rishi Saket. "Inapproximability of Minimum Vertex Cover on k-Uniform k-Partite Hypergraphs". In: *SIAM J. Discret. Math.* 29.1 (2015), pp. 36–58.

[Guo+12] Zhenyu Guo, Xuepeng Fan, Rishan Chen, Jiaxing Zhang, Hucheng Zhou, Sean McDirmid, Chang Liu, Wei Lin, Jingren Zhou, and Lidong Zhou. "Spotting code optimizations in data-parallel pipelines through PeriSCOPE". In: *Presented as part of the 10th USENIX Symposium on Operating Systems Design and Implementation (OSDI 12)*. 2012, pp. 121–133.

[GZ12] Venkatesan Guruswami and Yuan Zhou. "Tight Bounds on the Approximability of Almost-Satisfiable Horn SAT and Exact Hitting Set". In: *Theory Comput.* 8.1 (2012), pp. 239–267.

[Hås01] Johan Håstad. "Some optimal inapproximability results". In: *J. ACM* 48.4 (2001), pp. 798–859.

[Hax99] Penny E. Haxell. "Packing and covering triangles in graphs". In: *Discret. Math.* 195.1-3 (1999), pp. 251–254.

[HCG12] Chi-Yao Hong, Matthew Caesar, and P Brighten Godfrey. "Finishing flows quickly with preemptive scheduling". In: *ACM SIGCOMM Computer Communication Review* 42.4 (2012), pp. 127–138.

[HLV94] J. A. Hoogeveen, Jan Karel Lenstra, and Bart Veltman. "Three, four, five, six, or the complexity of scheduling with communication delays". In: *Oper. Res. Lett.* 16.3 (1994), pp. 129–137.

[HM01] C. Hanen and A. Munier. "An approximation algorithm for scheduling dependent tasks on m processors with small communication delays". In: *Discrete Applied Mathematics* 108.3 (2001), pp. 239–257.

[Hol02] Jonas Holmerin. "Vertex cover on 4-regular hyper-graphs is hard to approximate within 2-epsilon". In: *Proceedings on 34th Annual ACM Symposium on Theory of Computing*. 2002, pp. 544–552.

[HR01] Penny E. Haxell and Vojtech Rödl. "Integer and Fractional Packings in Dense Graphs". In: *Comb.* 21.1 (2001), pp. 13–38.

[HS19] David G. Harris and Aravind Srinivasan. "The Moser-Tardos Framework with Partial Resampling". In: *J. ACM* 66.5 (2019), 36:1–36:45.

[HS87] Dorit S. Hochbaum and David B. Shmoys. "Using dual approximation algorithms for scheduling problems theoretical and practical results". In: *J. ACM* 34.1 (1987), pp. 144–162.

[HSS06] Elad Hazan, Shmuel Safra, and Oded Schwartz. "On the complexity of approximating k-set packing". In: *Computational Complexity* 15.1 (2006), pp. 20–39.

[Im+19] Sungjin Im, Nathaniel Kell, Janardhan Kulkarni, and Debmalya Panigrahi. "Tight Bounds for Online Vector Scheduling". In: *SIAM J. Comput.* 48.1 (2019), pp. 93–121.

[JCG97] Peter Jeavons, David A. Cohen, and Marc Gyssens. "Closure properties of constraints". In: *J. ACM* 44.4 (1997), pp. 527–548.

[Jea98] Peter Jeavons. "On the Algebraic Structure of Combinatorial Problems". In: *Theor. Comput. Sci.* 200.1-2 (1998), pp. 185–204.

[Juh82] Ferenc Juhász. "The asymptotic behaviour of Lovász' theta-function for random graphs". In: *Comb.* 2.2 (1982), pp. 153–155.

[JZA19] Zhihao Jia, Matei Zaharia, and Alex Aiken. "Beyond Data and Model Parallelism for Deep Neural Networks". In: *Proceedings of the 2nd SysML Conference, SysML '19*. Palo Alto, CA, USA, 2019.

[Kal04] Gil Kalai. "Social Indeterminacy". In: *Econometrica* 72.5 (2004), pp. 1565–1581.

[Kan+21] Dong Yeap Kang, Tom Kelly, Daniela Kühn, Abhishek Methuku, and Deryk Osthus. *A proof of the Erdős-Faber-Lovász conjecture*. 2021. arXiv: `2101.04698 [math.CO]`.

[KAR00] V. S. Anil Kumar, Sunil Arya, and H. Ramesh. "Hardness of Set Cover with Intersection 1". In: *Automata, Languages and Programming, 27th International Colloquium, ICALP 2000, Geneva, Switzerland, July 9-15, 2000, Proceedings*. Vol. 1853. Lecture Notes in Computer Science. Springer, 2000, pp. 624–635.

[Kho+07] Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. "Optimal Inapproximability Results for MAX-CUT and Other 2-Variable CSPs?" In: *SIAM J. Comput.* 37.1 (2007), pp. 319–357.

[Kho01] Subhash Khot. "Improved Inaproximability Results for MaxClique, Chromatic Number and Approximate Graph Coloring". In: *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001*. 2001, pp. 600–609.

[Kho02a] Subhash Khot. "Hardness results for coloring 3-colorable 3-uniform hypergraphs". In: *The 43rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2002*. IEEE. 2002, pp. 23–32.

[Kho02b] Subhash Khot. "On the power of unique 2-prover 1-round games". In: *Proceedings on 34th Annual ACM Symposium on Theory of Computing, STOC 2002*. ACM, 2002, pp. 767–775.

[KMS17] Subhash Khot, Dor Minzer, and Muli Safra. "On independent sets, 2-to-2 games, and Grassmann graphs". In: *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017,* ACM, 2017, pp. 576–589.

[KMS18] Subhash Khot, Dor Minzer, and Muli Safra. "Pseudorandom Sets in Grassmann Graph Have Near-Perfect Expansion". In: *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018*. IEEE Computer Society, 2018, pp. 592–601.

[Knu94] Donald E. Knuth. "The Sandwich Theorem". In: *Electron. J. Comb.* 1 (1994).

[KO19] Andrei A. Krokhin and Jakub Opršal. "The Complexity of 3-Colouring H-Colourable Graphs". In: *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019*. IEEE Computer Society, 2019, pp. 1227–1239.

[KP20] Jeff Kahn and Jinyoung Park. *Tuza's Conjecture for random graphs*. 2020. arXiv: `2007.04351 [math.CO]`.

[KR83] K. H. Kim and F. W. Roush. "On a problem of Turán". In: *Studies in Pure Mathematics: To the Memory of Paul Turán*. Basel: Birkhäuser Basel, 1983, pp. 423–425.

[Kri95] Michael Krivelevich. "On a conjecture of Tuza about packing and covering of triangles". In: *Discret. Math.* 142.1-3 (1995), pp. 281–286.

[Kro+20] Andrei A. Krokhin, Jakub Opršal, Marcin Wrochna, and Stanislav Živný. "Topology and adjunction in promise constraint satisfaction". In: *Electron. Colloquium Comput. Complex.* (2020), p. 40.

[KS14] Subhash Khot and Rishi Saket. "Hardness of Finding Independent Sets in 2-Colorable and Almost 2-Colorable Hypergraphs". In: *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms*. 2014, pp. 1607–1625.

[KT17] Ken-Ichi Kawarabayashi and Mikkel Thorup. "Coloring 3-Colorable Graphs with Less Than $n^{1/5}$ Colors". In: *J. ACM* 64.1 (Mar. 2017).

[Kul+20] Janardhan Kulkarni, Shi Li, Jakub Tarnawski, and Minwei Ye. "Hierarchy-Based Algorithms for Minimizing Makespan under Precedence and Communication Constraints". In: *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*. SIAM, 2020, pp. 2770–2789.

[KZ97] Howard J. Karloff and Uri Zwick. "A 7/8-Approximation Algorithm for MAX 3SAT?" In: *38th Annual Symposium on Foundations of Computer Science, FOCS '97, Miami Beach, Florida, USA, October 19-22, 1997*. IEEE Computer Society, 1997, pp. 406–415.

[Law+93] Eugene L Lawler, Jan Karel Lenstra, Alexander HG Rinnooy Kan, and David B Shmoys. "Sequencing and scheduling: Algorithms and complexity". In: *Handbooks in operations research and management science* 4 (1993), pp. 445–522.

[Lee19] Euiwoong Lee. "Partitioning a graph into small pieces with applications to path transversal". In: *Math. Program.* 177.1-2 (2019), pp. 1–19.

[Li21] Shi Li. “Towards PTAS for Precedence Constrained Scheduling via Combinatorial Algorithms”. In: *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM. 2021, pp. 2991–3010.

[LMR94] Frank Thomson Leighton, Bruce M. Maggs, and Satish Rao. “Packet Routing and Job-Shop Scheduling in O(Congestion + Dilation) Steps”. In: *Comb.* 14.2 (1994), pp. 167–186.

[Lov75] Láslzó Lovász. “On minmax theorems of combinatorics, Doctoral thesis”. In: *Mathematiki Lapok* 26 (1975).

[LR16] E. Levey and T. Rothvoss. “A (1+epsilon)-approximation for makespan scheduling with precedence constraints using LP hierarchies”. In: *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*. 2016, pp. 168–177.

[Luo+16] Shouxi Luo, Hongfang Yu, Yangming Zhao, Sheng Wang, Shui Yu, and Lemin Li. “Towards practical and near-optimal coflow scheduling for data center networks”. In: *IEEE Transactions on Parallel and Distributed Systems* 27.11 (2016), pp. 3366–3380.

[LY94] Carsten Lund and Mihalis Yannakakis. “On the Hardness of Approximating Minimization Problems”. In: *J. ACM* 41.5 (1994), pp. 960–981.

[LZ09] Linyuan Lu and Yi Zhao. “An Exact Result for Hypergraphs and Upper Bounds for the Turán Density of K^r_{r+1}”. In: *SIAM J. Discret. Math.* 23.3 (2009), pp. 1324–1334.

[Mai+20] Biswaroop Maiti, Rajmohan Rajaraman, David Stalfa, Zoya Svitkina, and Aravindan Vijayaraghavan. “Scheduling Precedence-Constrained Jobs on Related Machines with Communication Delay”. In: *To appear at 61st Annual IEEE Symposium on Foundations of Computer Science (FOCS 2020)* (2020).

[Mar74] G. A. Margulis. “Probabilistic characteristics of graphs with large connectivity (in Russian)”. In: *Probl. Pered. Inform.* 10 (1974), pp. 101–108.

[McD93] Colin McDiarmid. “A Random Recolouring Method for Graphs and Hypergraphs”. In: *Combinatorics, Probability and Computing* 2.3 (1993), pp. 363–365.

[Mic+13] Tomasz P. Michalak, Aadithya V. Karthik, Piotr L. Szczepanski, Balaraman Ravindran, and Nicholas R. Jennings. “Efficient Computation of the Shapley Value for Game-Theoretic Network Centrality”. In: *J. Artif. Intell. Res.* 46 (2013), pp. 607–650.

[Mir+17] Azalia Mirhoseini, Hieu Pham, Quoc V Le, Benoit Steiner, Rasmus Larsen, Yuefeng Zhou, Naveen Kumar, Mohammad Norouzi, Samy Bengio, and Jeff Dean. “Device placement optimization with reinforcement learning”. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 2430–2439.

[MK97] A. Munier and J.C. König. “A Heuristic for a Scheduling Problem with Communication Delays”. In: *Operations Research* 45.1 (1997), pp. 145–147.

[MOO10] Elchanan Mossel, Ryan O’Donnell, and Krzysztof Oleszkiewicz. “Noise stability of functions with low influences: invariance and optimality”. In: *Annals of Mathematics* 171 (2010), pp. 295–341.

[Mos10] Elchanan Mossel. “Gaussian Bounds for Noise Correlation of Functions”. In: *Geometric and Functional Analysis* 19 (2010), pp. 1713–1756.

[Mos15] Dana Moshkovitz. “The Projection Games Conjecture and the NP-Hardness of ln n-Approximating Set-Cover”. In: *Theory Comput.* 11 (2015), pp. 221–235.

[MR10] Dana Moshkovitz and Ran Raz. “Two-query PCP with subconstant error”. In: *J. ACM* 57.5 (2010), 29:1–29:29.

[MR95] Jiří Matoušek and Vojtěch Rödl. “On Ramsey sets in spheres”. In: *Journal of Combinatorial Theory, Series A* 70.1 (1995), pp. 30–44.

[MRT13] Adam Meyerson, Alan Roytman, and Brian Tagiku. “Online Multidimensional Load Balancing”. In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2013*. 2013, pp. 287–302.

[MS11] Monaldo Mastrolilli and Ola Svensson. “Hardness of Approximating Flow and Job Shop Scheduling Problems”. In: *J. ACM* 58.5 (2011), 20:1–20:32.

[Nar+19] D. Narayanan, A. Harlap, A. Phanishayee, V. Seshadri, N. Devanur, G. Ganger, P. Gibbons, and M. Zaharia. “PipeDream: Generalized Pipeline Parallelism for DNN Training”. In: *Proc. 27th ACM Symposium on Operating Systems Principles (SOSP)*. Huntsville, ON, Canada, Oct. 2019.

[Nis89] N. Nisan. “CREW PRAMs and Decision Trees”. In: *Proceedings of the Twenty-first Annual ACM Symposium on Theory of Computing*. STOC ’89. ACM, 1989, pp. 327–335.

[NN11] Ramasuri Narayanam and Yadati Narahari. “A Shapley Value-Based Approach to Discover Influential Nodes in Social Networks”. In: *IEEE Trans Autom. Sci. Eng.* 8.1 (2011), pp. 130–147.

[ODo14] Ryan O’Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014.

[OW09] Ryan O’Donnell and Yi Wu. “Conditional hardness for satisfiable 3-CSPs”. In: *Proceedings of the 41st Annual ACM Symposium on Theory of Computing*. 2009, pp. 493–502.

[Pan+11] Rina Panigrahy, Kunal Talwar, Lincoln Uyeda, and Udi Wieder. “Heuristics for Vector Bin Packing”. 2011. URL: `https://www.microsoft.com/en-us/research/publication/heuristics-for-vector-bin-packing/`.

[Pet20] Jan Petr. *Monotone functions avoiding majorities*. Undergraduate Thesis. Univerzita Karlova, Matematicko-fyzikalni fakulta. 2020.

[Pip02a] Nicholas Pippenger. “Galois theory for minors of finite functions”. In: *Discrete Mathematics* 254.1-3 (2002), pp. 405–419.

[Pip02b] Nicholas Pippenger. “Galois theory for minors of finite functions”. In: *Discrete Mathematics* 254.1 (2002), pp. 405–419.

[Pol12] D.H.J. Polymath. “A new proof of the density Hales-Jewett theorem”. In: *Annals of Mathematics* 175.3 (May 2012), pp. 1283–1327.

[PR81] S. Poljak and V. Rödl. “On the arc-chromatic number of a digraph”. In: *Journal of Combinatorial Theory, Series B* 31.2 (1981), pp. 190–198.

[Pul15] Gregory J. Puleo. "Tuza's Conjecture for graphs with maximum average degree less than 7". In: *Eur. J. Comb.* 49 (2015), pp. 134–152.

[PY90] Christos H. Papadimitriou and Mihalis Yannakakis. "Towards an Architecture-Independent Analysis of Parallel Algorithms". In: *SIAM J. Comput.* 19.2 (1990), pp. 322–328.

[Rag08] Prasad Raghavendra. "Optimal algorithms and inapproximability results for every CSP?" In: *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*. ACM, 2008, pp. 245–254.

[Ray21] Arka Ray. "There is no APTAS for 2-dimensional vector bin packing: Revisited". In: *CoRR* abs/2104.13362 (2021). arXiv: `2104.13362`.

[Ray87] Victor J. Rayward-Smith. "UET scheduling with unit interprocessor communication delays". In: *Discret. Appl. Math.* 18.1 (1987), pp. 55–71.

[Raz98] Ran Raz. "A Parallel Repetition Theorem". In: *SIAM J. Comput.* 27.3 (1998), pp. 763–803.

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

[Rus82] Lucio Russo. "An approximate zero-one law". In: *Z. Wahrscheinlichkeitstheorie und Verwandte Gebiete* 61.1 (1982), pp. 129–139.

[Sak14] Rishi Saket. "Hardness of Finding Independent Sets in 2-Colorable Hypergraphs and of Satisfiable CSPs". In: *Proceedings of the 29th IEEE Conference on Computational Complexity*. 2014, pp. 78–89.

[San21] Sai Sandeep. "Almost Optimal Inapproximability of Multidimensional Packing Problems". In: *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*. IEEE, 2021, pp. 245–256.

[Sch78] Thomas J. Schaefer. "The Complexity of Satisfiability Problems". In: *Proceedings of the 10th Annual ACM Symposium on Theory of Computing, STOC 1978*. ACM, 1978, pp. 216–226.

[Shy+18] Ayan Shymyrbay, Arshyn Zhanbolatov, Assilkhan Amankhan, Adilya Bakambekova, and Ikechi A Ukaegbu. "Meeting Deadlines in Datacenter Networks: An Analysis on Deadline-Aware Transport Layer Protocols". In: *2018 International Conference on Computing and Network Communications (CoCoNet)*. IEEE. 2018, pp. 152–158.

[Sid95] Alexander Sidorenko. "What we know and what we do not know about Turán numbers". In: *Graphs Comb.* 11.2 (1995), pp. 179–199.

[Sid97] Alexander Sidorenko. "Upper Bounds for Turán Numbers". In: *J. Comb. Theory, Ser. A* 77.1 (1997), pp. 134–147.

[Sim83] Hans-Ulrich Simon. "A tight Ω(loglog n)-bound on the time for parallel RAMs to compute nondegenerated boolean functions". In: *Foundations of Computation Theory*. Springer Berlin Heidelberg, 1983, pp. 439–444.

[SK67] Richard Sinkhorn and Paul Knopp. "Concerning nonnegative matrices and doubly stochastic matrices". In: *Pacific Journal of Mathematics* 21.2 (1967), pp. 343–348.

[Spi94] Frits CR Spieksma. “A branch-and-bound algorithm for the two-dimensional vector packing problem”. In: *Computers & operations research* 21.1 (1994), pp. 19–25.

[SS13] Sushant Sachdeva and Rishi Saket. “Optimal Inapproximability for Scheduling Problems via Structural Hardness for Hypergraph Vertex Cover”. In: *Proceedings of the 28th Conference on Computational Complexity, CCC 2013, K.lo Alto, California, USA, 5-7 June, 2013*. 2013, pp. 219–229.

[SS54] L. S. Shapley and Martin Shubik. “A Method for Evaluating the Distribution of Power in a Committee System”. In: *American Political Science Review* 48.3 (1954), pp. 787–792.

[SSW94] David B. Shmoys, Clifford Stein, and Joel Wein. “Improved Approximation Algorithms for Shop Scheduling Problems”. In: *SIAM J. Comput.* 23.3 (1994), pp. 617–632.

[ST12] Hadas Shachnai and Tami Tamir. “Approximation schemes for generalized two-dimensional vector packing with application to data placement”. In: *J. Discrete Algorithms* 10 (2012), pp. 35–48.

[Sve10] Ola Svensson. “Conditional hardness of precedence constrained scheduling on identical machines”. In: *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*. ACM, 2010, pp. 745–754.

[SW99a] P. Schuurman and G. Woeginger. “Polynomial time approximation algorithms for machine scheduling: ten open problems”. In: *Journal of Scheduling* 2.5 (1999), pp. 203–213.

[SW99b] P. Schuurman and G. J. Woeginger. *Polynomial time approximation algorithms for machine scheduling: Ten open problems*. 1999.

[Tar+20] Jakub Tarnawski, Amar Phanishayee, Nikhil R. Devanur, Divya Mahajan, and Fanny Nina Paravecino. *Efficient Algorithms for Device Placement of DNN Graph Operators*. 2020. arXiv: `2006.16423 [cs.LG]`.

[Tre01] Luca Trevisan. “Non-approximability results for optimization problems on bounded degree instances”. In: *Proceedings on 33rd Annual ACM Symposium on Theory of Computing, 2001*. 2001, pp. 453–461.

[Tur41] Paul Turán. “On an extremal problem in graph theory”. In: *Mat. Fiz. Lapok* 48 (1941), pp. 436–452.

[Tur61] Paul Turán. “Research Problem”. In: *Közl MTA Mat. Kutató Int.* 6 (1961), pp. 417–423.

[Tuz81] Zsolt Tuza. “Conjecture”. In: *Finite and Infinite Sets* Proc. Colloq. Math. Soc. Janos Bolyai (1981), p. 888.

[Tuz90] Zsolt Tuza. “A conjecture on triangles of graphs”. In: *Graphs Comb.* 6.4 (1990), pp. 373–380.

[TY92] R. Thurimella and Y. Yesha. “A scheduling principle for precedence graphs with communication delay”. In: *International Conference on Parallel Processing* 3 (1992), pp. 229–236.

[TŽ18] Johan Thapper and Stanislav Živný. “The Limits of SDP Relaxations for General-Valued CSPs”. In: *ACM Trans. Comput. Theory* 10.3 (2018), 12:1–12:22.

[VL81] Wenceslas Fernandez de la Vega and George S. Lueker. “Bin packing can be solved within 1+epsilon in linear time”. In: *Comb.* 1.4 (1981), pp. 349–355.

[VLL90] B. Veltman, B.J. Lageweg, and J.K. Lenstra. “Multiprocessor scheduling with communication delays”. In: *Parallel Computing* 16.2 (1990), pp. 173–182.

[Web77] Robert Weber. *Probabilistic Values for Games*. Cowles Foundation Discussion Papers 471R. Cowles Foundation for Research in Economics, Yale University, 1977.

[Wen13] Cenny Wenner. “Circumventing d-to-1 for Approximation Resistance of Satisfiable Predicates Strictly Containing Parity of Width at Least Four”. In: *Theory of Computing* 9.23 (2013), pp. 703–757.

[Woe97] Gerhard J. Woeginger. “There is no Asymptotic PTAS for Two-Dimensional Vector Packing”. In: *Inf. Process. Lett.* 64.6 (1997), pp. 293–297.

[WS11] David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms*. USA: Cambridge University Press, 2011.

[WŽ20] Marcin Wrochna and Stanislav Živný. “Improved hardness for H-colourings of G-colourable graphs”. In: *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020*. SIAM, 2020, pp. 1426–1435.

[Yus12] Raphael Yuster. “Dense Graphs With a Large Triangle Cover Have a Large Triangle Packing”. In: *Comb. Probab. Comput.* 21.6 (2012), pp. 952–962.

[Zha+12] Jiaxing Zhang, Hucheng Zhou, Rishan Chen, Xuepeng Fan, Zhenyu Guo, Haoxiang Lin, Jack Y Li, Wei Lin, Jingren Zhou, and Lidong Zhou. “Optimizing data shuffling in data-parallel computation by understanding user-defined functions”. In: *Presented as part of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12)*. 2012, pp. 295–308.

[Zha+15] Yangming Zhao, Kai Chen, Wei Bai, Minlan Yu, Chen Tian, Yanhui Geng, Yiming Zhang, Dan Li, and Sheng Wang. “Rapier: Integrating routing and scheduling for coflow-aware data center networks”. In: *2015 IEEE Conference on Computer Communications (INFOCOM)*. IEEE. 2015, pp. 424–432.

[Zhu20] Dmitriy Zhuk. “A Proof of the CSP Dichotomy Conjecture”. In: *J. ACM* 67.5 (2020), 30:1–30:78.

[Zwi98] Uri Zwick. “Finding Almost-Satisfying Assignments”. In: *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*. Ed. by Jeffrey Scott Vitter. ACM, 1998, pp. 551–560.