

# The Average-Case Complexity of Counting Cliques in Erdős-Rényi Hypergraphs

by

Enric Boix

A.B., Princeton University (2018)

Submitted to the Department of Electrical Engineering and Computer  
Science

in partial fulfillment of the requirements for the degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2020

© Massachusetts Institute of Technology 2020. All rights reserved.

Author .....  
Department of Electrical Engineering and Computer Science  
August 28, 2020

Certified by .....  
Guy Bresler  
Associate Professor of Electrical Engineering and Computer Science  
Thesis Supervisor

Accepted by .....  
Leslie A. Kolodziejski  
Professor of Electrical Engineering and Computer Science  
Chair, Department Committee on Graduate Students



# The Average-Case Complexity of Counting Cliques in Erdős-Rényi Hypergraphs

by

Enric Boix

Submitted to the Department of Electrical Engineering and Computer Science  
on August 28, 2020, in partial fulfillment of the  
requirements for the degree of  
Master of Science

## Abstract

The complexity of clique problems on Erdős-Rényi random graphs has become a central topic in average-case complexity. Algorithmic phase transitions in these problems have been shown to have broad connections ranging from mixing of Markov chains and statistical physics to information-computation gaps in high-dimensional statistics. We consider the problem of counting  $k$ -cliques in  $s$ -uniform Erdős-Rényi hypergraphs  $G(n, c, s)$  with edge density  $c$  and show that its fine-grained average-case complexity can be based on its worst-case complexity. We give a worst-case to average-case reduction for counting  $k$ -cliques on worst-case hypergraphs given a blackbox solving the problem on  $G(n, c, s)$  with low error probability. Our approach is closely related to [Goldreich and Rothblum, FOCS18], which showed a worst-case to average-case reduction for counting cliques for an efficiently-sampleable distribution on graphs.

Our reduction has the following implications:

- *Dense Erdős-Rényi graphs and hypergraphs:* Counting  $k$ -cliques on  $G(n, c, s)$  with  $k$  and  $c$  constant matches its worst-case complexity up to a polylog( $n$ ) factor. Assuming ETH, it takes  $n^{\Omega(k)}$  time to count  $k$ -cliques in  $G(n, c, s)$  if  $k$  and  $c$  are constant.
- *Sparse Erdős-Rényi graphs and hypergraphs:* When  $c = \Theta(n^{-\alpha})$ , for each fixed  $\alpha$  our reduction yields different average-case phase diagrams depicting a tradeoff between runtime and  $k$ . Assuming the best known worst-case algorithms are optimal, in the graph case of  $s = 2$ , we establish that the exponent in  $n$  of the optimal running time for  $k$ -clique counting in  $G(n, c, s)$  is  $\frac{\omega k}{3} - C\alpha \binom{k}{2} + O_{k,\alpha}(1)$ , where  $\frac{\omega}{9} \leq C \leq 1$  and  $\omega$  is the matrix multiplication constant. In the hypergraph case of  $s \geq 3$ , we show a lower bound at the exponent of  $k - \alpha \binom{k}{s} + O_{k,\alpha}(1)$  which surprisingly is tight against algorithmic achievability exactly for the set of  $c$  above the Erdős-Rényi  $k$ -clique percolation threshold.

Our reduction yields the first average-case hardness result for a problem over Erdős-Rényi hypergraphs based on a corresponding worst-case hardness assumption. More-

over, because we consider sparse Erdős-Rényi hypergraphs, for each  $n, k$ , and  $s$  we actually have an entire family of problems parametrized by the edge probability  $c$  and the behavior changes as a function of  $c$ ; this is the first worst-to-average-case hardness result we are aware of for which the complexity of the same problem over worst-case versus average-case inputs is completely different. We also analyze several natural algorithms for counting  $k$ -cliques in  $G(n, c, s)$  that establish our upper bounds in the sparse case  $c = \Theta(n^{-\alpha})$ . The ingredients in our worst-case to average-case reduction include: (1) several new techniques for the self-reducibility of counting  $k$ -cliques as a low-degree polynomial; and (2) a finite Fourier analytic method to bound the total variation convergence of random biased binary expansions to the uniform distribution over residues in  $\mathbb{F}_p$ .

Thesis Supervisor: Guy Bresler

Title: Associate Professor of Electrical Engineering and Computer Science

## Acknowledgments

My greatest appreciation goes to Guy Bresler, who has been a wonderful advisor over the last two years. Throughout, he has unfailingly supported me in my development as a researcher, optimistically encouraged me to explore even the wildest ideas, and constantly inspired me with his fantastic taste in problems. I would also like to express my gratitude to my incredible coauthor, Matt Brennan, without whom this master's thesis would not have been written. Matt Brennan, Austin Stromme, Dheeraj Nagaraj, Fred Koehler, Rares Buhai, and Sinho Chewi have been the funniest, the most nurturing, and quite simply the best group of office-mates that I could have hoped for. I'm looking forward to many more late-night conversations once we all get back on campus. I would like to thank my other coauthors in grad school, Ben Edelman, Emmanuel Abbe, Jason Altschuler, Lior Eldar, Saeed Mehraban, and Siddhartha Jayanti, for their friendship and their tenacity in tackling problems. I am also greatly indebted to the MIT community as a whole, for being the fount of many fantastic friendships. A special thank you goes to Sonia, Smita, and Imran for their warm hospitality during the writing of this thesis. Finally, my parents, brothers, and grandmother have always been a source of comfort and love.

And, last but not least: much of the material in this thesis is reprinted, with permission, from [14] ©2019 IEEE.



# Contents

<b>1</b>	<b>Introduction</b>	<b>11</b>
1.1	Related Work . . . . .	14
1.1.1	Clique Problems on Erdős-Rényi Graphs . . . . .	14
1.1.2	Worst-Case to Average-Case Reductions . . . . .	15
1.1.3	Comparison to Goldreich-Rothblum '18 . . . . .	16
1.1.4	Subsequent Work . . . . .	17
1.2	Overview of Main Results . . . . .	17
1.3	Overview of Reduction Techniques . . . . .	21
1.4	Notation and Preliminaries . . . . .	24
1.5	Organization . . . . .	24
<b>2</b>	<b>Problem Formulations and Average-Case Lower Bounds</b>	<b>27</b>
2.1	Clique Problems and Worst-Case Fine-Grained Conjectures . . . . .	27
2.2	Average-Case Lower Bounds for Counting $k$ -Cliques in $G(n, c, s)$ . . . . .	30
<b>3</b>	<b>Worst-Case to Average-Case Reduction for <math>G(n, c, s)</math></b>	<b>37</b>
3.1	Worst-Case Reduction to $k$ -Partite Hypergraphs . . . . .	39
3.2	Counting $k$ -Cliques as a Low-Degree Polynomial . . . . .	40
3.3	Random Self-Reducibility: Reducing to Random Inputs in $\mathbb{F}^N$ . . . . .	41
3.4	Reduction to Evaluating the Polynomial on $G(nk, c, s, k)$ . . . . .	42
3.5	Reduction to Counting $k$ -Cliques in $G(n, c, s)$ . . . . .	47
3.6	Proofs of Theorems 2.2.1 and 2.2.2 . . . . .	48

<b>4</b>	<b>Random Binary Expansions Modulo <math>p</math></b>	<b>55</b>
<b>5</b>	<b>Algorithms for Counting <math>k</math>-Cliques in <math>G(n, c, s)</math></b>	<b>63</b>
5.1	Greedy Random Sampling . . . . .	63
5.2	Sample Complexity and Runtime of Greedy Random Sampling . . . .	65
5.3	Post-Processing with Matrix Multiplication . . . . .	70
5.4	Deterministic Iterative Algorithm for Counting in $G(n, c, s)$ . . . . .	73
<b>6</b>	<b>Extensions and Open Problems</b>	<b>75</b>



# List of Figures

- 1-1 Comparison of our algorithms and average-case lower bounds for counting  $k$ -cliques in sparse Erdős-Rényi Hypergraphs  $G(n, c, s)$  with  $c = \Theta(n^{-\alpha})$ . Green denotes runtimes  $T$  feasible for each  $k$ , blue denotes  $T$  infeasible given that the best known worst-case algorithms are optimal and gray denotes  $T$  for which the complexity of counting  $k$ -cliques is open after this work. The left plot shows the graph case of  $s = 2$  and the right plot shows the hypergraph case of  $s \geq 3$ . For simplicity, all quantities shown are up to constant  $O_{k,\alpha}(1)$  additive error. ©2019 IEEE. 20
- 3-1 Reduction TO-ER-# for showing computational lower bounds for average-case  $\#(k, s)$ -CLIQUE on Erdős-Rényi  $G(n, c, s)$  hypergraphs based on the worst-case hardness of  $\#(k, s)$ -CLIQUE. ©2019 IEEE. . . . . 54



# Chapter 1

## Introduction

A graph  $G = (V, E)$  consists of a set of  $n$  vertices,  $V$ , and a set of edges,  $E \subseteq \binom{V}{2}$ . A  $k$ -clique in  $G$  is a subset  $S \subseteq V$  of  $k$  pairwise-adjacent vertices, i.e.,  $\binom{S}{2} \subseteq E$ .

We answer the following fundamental question: is it harder to count the number of  $k$ -cliques in an adversarially-chosen graph than in a uniformly-random graph? As we discuss below, this question lies at the intersection of fine-grained complexity and average-case complexity.

**Fine-grained complexity** In our analysis, we will think of  $k$  as a small constant. Thus, the naive brute-force algorithm that enumerates  $k$ -vertex subsets and checks each one takes  $\Theta(k^2 \binom{n}{k}) = \Theta(n^k) = \text{poly}(n)$  time. This runtime is polynomial in  $n$ , so, from the point of view of *classical* complexity theory, the brute-force algorithm is efficient and the case is closed.

However, the story does not quite end here: the exponential dependence in the parameter  $k$  is highly unsatisfying because it means that brute force is impractical even for moderate values of  $k$ . A *fine-grained* complexity theorist would ask whether faster algorithms are possible [77]. Is the exponential dependence in  $k$  unavoidable? What precisely is the best runtime that one can achieve for counting  $k$ -cliques? The answer, it turns out, is that if the graph is adversarially chosen then a  $n^{\Omega(k)}$  worst-case runtime is unavoidable<sup>1</sup> [20].

---

<sup>1</sup>Under standard complexity theory assumptions.

**Average-case complexity** However, we wish to understand how hard it is to count  $k$ -cliques when the graph is not adversarially chosen, but rather is chosen uniformly at random. Is  $n^{\Omega(k)}$  time still necessary, or can we speed up by exploiting the random graph structure?

This is a question in *average-case* complexity, as opposed to classical *worst-case* complexity, because the algorithm is not required to work for all possible inputs – just for typical inputs drawn from a distribution [58]. The average-case and worst-case complexities of a problem can differ wildly: sometimes solving a problem on a typical random input is much easier than solving it on any, potentially adversarially-chosen input. This phenomenon can be used to explain, for instance, why modern SAT solvers can efficiently solve constraint satisfaction instances that arise in practice, even though constraint satisfaction would be intractable if the instances were adversarially crafted. On the other hand, many problems are conjectured to be intractable in both the worst-case and average-case settings – indeed, the existence of such problems is a necessary assumption of secure cryptography.

The average-case complexity of the  $k$ -clique counting on random graphs is of particular interest (i) because it is a candidate problem that could be used as a basis for constructions in the recently-minted field of fine-grained cryptography [8, 9, 45, 44, 57, 24] and (ii) because the study of clique problems on random graphs is known to have an intimate connection to the computational complexity of high-dimensional statistical inference [11, 55, 21, 47, 60, 17, 18, 16].

**Informal statement of results** A main result of this thesis is that, for any constant  $k$ , counting  $k$ -cliques in a uniformly random graph is just as hard as counting  $k$ -cliques in an adversarially-chosen graph.

More generally, we consider the average-case complexity of counting  $k$ -cliques in  $s$ -uniform Erdős-Rényi hypergraphs  $G(n, c, s)$ , where every  $s$ -subset of the  $n$  vertices is a hyperedge independently with probability  $c$ . We reduce counting  $k$ -cliques on worst-case hypergraphs to a blackbox solving the problem on  $G(n, c, s)$  with low error probability. Our reduction yields different average-case lower bounds for counting  $k$ -

cliques in  $G(n, c, s)$  in the dense and sparse cases of  $c = \Theta(1)$  and  $c = \Theta(n^{-\alpha})$ , with tradeoffs between runtime and  $c$ , based on the worst-case complexity of counting  $k$ -cliques. We also derive improved algorithms for counting  $k$ -cliques on sparse  $G(n, c, s)$  and show that our average-case lower bounds often match algorithmic upper bounds.

**Main proof ideas** *Worst-case-to-average-case reduction:* Our reduction is based on the classical random self-reducibility of low-degree polynomials [59] over large finite fields  $\mathbb{F}$ . This has also recently been applied in other contexts in average-case fine-grained complexity [8, 44, 24]. The random self-reducibility of low-degree polynomials is the theorem that, given access to a blackbox  $\hat{P}(x)$  computing a degree- $d$  polynomial  $P(x)$  on most inputs  $x \in \mathbb{F}^n$ , we can “correct”  $\hat{P}$  and efficiently compute  $P(y)$  for any adversarial input  $y \in \mathbb{F}^n$ .

The original correction algorithm, due to Lipton [59], involves picking a random  $a \in \mathbb{F}^n$  and computing  $\hat{f}(1), \dots, \hat{f}(d+1)$  for  $\hat{f}(t) := \hat{P}(y + ta)$ . Assuming  $\mathbb{F}$  is large enough,  $y + ta \in \mathbb{F}^n$  is uniformly random for each  $t \in [d+1]$ . Thus, with high probability  $\hat{f}(t) = f(t) := P(y + ta)$  for each  $t \in [d+1]$ . Since  $f(t) = P(y)$  is a degree- $d$  polynomial in  $t$ , we can efficiently interpolate  $f(1), \dots, f(d+1)$  to compute  $f(0) = P(y)$ .

In order to apply this result to our case, we let  $P(x)$  be the function of the adjacency matrix that counts  $k$ -cliques. This is a low-degree polynomial, so random self-reduction applies. However, there is a large technical obstacle that we must overcome: the above trick only proves that  $P(x)$  is hard to compute on random elements in  $\mathbb{F}^n$  for some large field  $\mathbb{F}$ , but we would like hardness for vectors of independent Bernoulli variables in  $\{0, 1\}^n$ . The bulk of our proof is thus dedicated to appropriately converting from random field elements in  $\mathbb{F}$  to biased Bernoulli random variables in  $\{0, 1\}$ .

*Algorithms:* Our fast algorithms for  $k$ -clique counting in sparse Erdős-Rényi hypergraphs are based on an iterative greedy sampling procedure. In the graph case, we obtain an extra speedup by applying a fast matrix multiplication postprocessing trick similar to the algorithm of [64] for worst-case graphs.

## 1.1 Related Work

### 1.1.1 Clique Problems on Erdős-Rényi Graphs

The complexity of clique problems on Erdős-Rényi random graphs has become a central topic in average-case complexity, discrete probability and high-dimensional statistics. While the Erdős-Rényi random graph  $G(n, 1/2)$  contains cliques of size roughly  $2 \log_2 n$ , a longstanding open problem of Karp is to find a clique of size  $(1 + \epsilon) \log_2 n$  in polynomial time for some constant  $\epsilon > 0$  [54]. Natural polynomial time search algorithms and the Metropolis process find cliques of size approximately  $\log_2 n$  but not  $(1 + \epsilon) \log_2 n$  [54, 46, 51, 61, 66]. A related line of research shows that local algorithms fail to find independent sets of size  $(1 + \epsilon)n \ln(d)/d$  in several random graph models with average degree  $d$  similar to Erdős-Rényi, even though the largest independent set has size roughly  $2n \ln(d)/d$  [39, 23, 67]. In [32], it is shown that any algorithm probing  $n^{2-\delta}$  edges of  $G(n, 1/2)$  in  $\ell$  rounds finds cliques of size at most  $(2 - \epsilon) \log_2 n$ .

A large body of work has considered planted clique (PC), the problem of finding a  $k$ -clique randomly planted in  $G(n, 1/2)$ . Since its introduction in [56] and [51], a number of spectral algorithms, approximate message passing, semidefinite programming, nuclear norm minimization and several other polynomial-time combinatorial approaches have been proposed and all appear to fail to recover the planted clique when  $k = o(\sqrt{n})$  [3, 33, 62, 35, 5, 25, 28, 22]. It has been shown that cliques of size  $k = o(\sqrt{n})$  cannot be detected by the Metropolis process [51], low-degree sum of squares (SOS) relaxations [10] and statistical query algorithms [37]. Furthermore, the conjecture that PC with  $k = o(\sqrt{n})$  cannot be solved in polynomial time has been used as an average-case assumption in cryptography [52]. An emerging line of work also shows that the PC conjecture implies a number of tight statistical-computational gaps, including in sparse PCA, community detection, universal submatrix detection, RIP certification and low-rank matrix completion [11, 55, 21, 47, 60, 17, 18, 16]. Recently, [6] also showed that super-polynomial length regular resolution is required to certify that Erdős-Rényi graphs do not contain cliques of size  $k = o(n^{1/4})$ .

Rossman [71, 72] has studied the classical  $k$ -CLIQUE decision problem on sparse Erdős-Rényi random graphs  $G \sim G(n, c)$  at the critical threshold  $c = \Theta(n^{-2/(k-1)})$ , where the existence of a  $k$ -clique occurs with probability bounded away from 0 and 1. The natural greedy algorithm that selects a random sequence of vertices  $v_1, v_2, \dots, v_t$  such that  $v_{i+1}$  is a random common neighbor of  $v_1, v_2, \dots, v_i$  can be shown to find a clique of size  $\lfloor (1 + \epsilon)k/2 \rfloor$  if repeated  $n^{\epsilon^2 k/4}$  times. This yields an  $O(n^{k/4 + O(1)})$  time algorithm for  $k$ -CLIQUE on  $G(n, c)$ . Rossman showed that bounded depth circuits solving  $k$ -CLIQUE on  $G(n, c)$  must have size  $\Omega(n^{k/4})$  in [71] and extended this lower bound to monotone circuits in [72]. A survey of this and related work can be found in [73].

All of the lower bounds for the clique problems on Erdős-Rényi random graphs above are against restricted classes of algorithms such as local algorithms, regular resolution, bounded-depth circuits, monotone circuits, the SOS hierarchy and statistical query algorithms. One reason for this is that there are general obstacles to basing average-case complexity on worst-case complexity. For example, natural approaches to polynomial-time worst-case to average-case reductions for NP-complete problems fail unless  $\text{coNP} \subseteq \text{NP/poly}$  [36, 13, 12]. The objective of this work is to show that this worst-case characterization of average-case complexity is possible in a fine-grained sense for the problem of counting  $k$ -cliques in  $s$ -uniform Erdős-Rényi hypergraphs  $G(n, c, s)$  with edge density  $c$ .

### 1.1.2 Worst-Case to Average-Case Reductions

The random self-reducibility of low-degree polynomials serves as the basis for several worst-case to average-case reductions found in the literature. One of the first applications of this method was to prove that the permanent is hard to evaluate on random inputs, even with polynomially-small probability of success, unless  $\text{P}^{\#\text{P}} = \text{BPP}$  [74, 19]. (Under the slightly stronger assumption that  $\text{P}^{\#\text{P}} \neq \text{AM}$ , and with different techniques, [34] proved that computing the permanent on large finite fields is hard even with exponentially small success probability.) Recently, [8] used the polynomial random self-reducibility result in the fine-grained setting in order to construct poly-

nomials that are hard to evaluate on most inputs, assuming fine-grained hardness conjectures for problems such as 3-SUM, ORTHOGONAL-VECTORS, and/or ALL-PAIRS-SHORTEST-PATHS. The random self-reducibility of polynomials was also used by Gamarnik [38] in order to prove that exactly computing the partition function of the Sherrington-Kirkpatrick model in statistical physics is hard on average.

If a problem is random self-reducible, then random instances of the problem are essentially as hard as worst-case instances, and therefore one may generate a hard instance of the problem by simply generating a random instance. Because of this, random self-reducibility plays an important role in cryptography: it allows one to base cryptographic security on random instances of a problem, which can generally be generated efficiently. A prominent example of a random-self reducible problem with applications to cryptography is the problem of finding a short vector in a lattice. In a seminal paper, Ajtai [1] gave a worst-case to average-case reduction for this short-vector problem. His ideas were subsequently applied to prove the average-case hardness of the Learning with Errors (LWE) problem, which underlies lattice cryptography [1, 70]. A good survey covering worst-case to average-case reductions in lattice cryptography is [69].

There are known restrictions on problems that are self-reducible. For example, non-adaptive worst-case to average-case reductions for NP-complete problems fail unless  $\text{coNP} \subseteq \text{NP/poly}$  [36, 13, 12].

### 1.1.3 Comparison to Goldreich-Rothblum '18

A motivating recent work by Goldreich and Rothblum [44] also considered worst-case to average-case reductions for  $k$ -clique counting. They provided such a reduction mapping to an efficiently sampleable distribution on graphs with a high min-entropy of  $\tilde{\Omega}(n^2)$ . In contrast to [44], our objectives are to: (1) map precisely to the natural distribution  $G(n, c, s)$  for different edge densities  $c$ , including  $c = \Theta(1)$  and the sparse case  $c = \Theta(n^{-\alpha})$ ; and (2) to characterize the tradeoff between the time-complexity of counting  $k$ -cliques in  $G(n, c, s)$  and the sparsity parameter  $\alpha$ . Achieving this requires new techniques for the self-reducibility of counting  $k$ -cliques as a low-degree



polynomial and a tight analysis of random biased binary expansions over  $\mathbb{F}_p$  with finite Fourier analysis. However, our techniques also come at the cost of requiring a low error probability ( $1/\text{polylog}(n)$  in the dense case and  $1/\text{poly}(n)$  in the sparse case) for the average-case blackbox solving  $k$ -clique counting on  $G(n, c, s)$ . This is in contrast to [44], where a very high error probability of  $1 - 1/\text{polylog}(n)$  is tolerated. It remains an interesting open problem to extend our results for  $G(n, c, s)$  to tolerate higher error blackboxes. This error tolerance and open problem are discussed further in Section 2.2 and Chapter 6, and how our techniques relate to those in [44] is discussed in Section 1.3 and Chapter 3.

### 1.1.4 Subsequent Work

Subsequent to the publication of our work, Goldreich [43] provided a simplified reduction for counting the parity of the number of cliques in the uniform  $G(n, 1/2)$  Erdős-Rényi graph case. Goldreich obtained error tolerance  $\exp(-k^2)$  in this case, which is an improvement over the error tolerance  $\exp(-\tilde{O}(k^2))$  in our Theorem 2.2.2.

Dalirrooyfard, Lincoln, and Vassilevska Williams [24] generalize our techniques to problems admitting what they call “good low-degree polynomials.” They apply this generalization to obtain average-case hardness for counting the number of copies of any graph  $H$  as an induced subgraph of an Erdős-Rényi graph  $G$ . Furthermore, they also prove that simple variations of the orthogonal vectors, 3-sum and zero-weight  $k$ -clique problems are hard to count on average for uniform inputs. They use these results to obtain average-case fine-grained hardness for various problems such as counting the number of regular expression matchings.

## 1.2 Overview of Main Results

We provide two complementary main results on the fine-grained average-case complexity of counting  $k$ -cliques in  $G(n, c, s)$ . The precise formulations of the problems we consider are in Section 2.1.

**Worst-case to average-case reduction.** We give a worst-case to average-case reduction from counting  $k$ -cliques in worst-case  $s$ -uniform hypergraphs to counting  $k$ -cliques in hypergraphs drawn from  $G(n, c, s)$ . This allows us to base the average-case fine-grained complexity of  $k$ -clique counting over Erdős-Rényi hypergraphs on its worst-case complexity, which can be summarized as follows. Counting  $k$ -cliques in worst-case hypergraphs is known to take  $n^{\Omega(k)}$  time assuming the Exponential Time Hypothesis (ETH)<sup>2</sup> if  $k = O(1)$  [20]. The best known worst-case algorithms up to subpolynomial factors are the  $O(n^{\omega \lceil k/3 \rceil})$  time algorithm of [64] in the graph case of  $s = 2$  and exhaustive  $O(n^k)$  time search on worst-case hypergraphs with  $s \geq 3$ . Here,  $\omega \approx 2.373$  denotes the best known matrix multiplication constant.

Our reduction is the first worst-case to average-case reduction to Erdős-Rényi hypergraphs. It has different implications for the cases of dense and sparse hypergraphs, as described next.

1. *Dense Erdős-Rényi Hypergraphs.* When  $k$  and  $c$  are constant, our reduction constructs an efficient  $k$ -clique counting algorithm that succeeds on a worst-case input hypergraph with high probability, using  $\text{polylog}(n)$  queries to an average-case oracle that correctly counts  $k$ -cliques on a  $1 - 1/\text{polylog}(n)$  fraction of Erdős-Rényi hypergraphs drawn from  $G(n, c, s)$ . This essentially shows that  $k$ -clique counting in the worst-case matches that on dense Erdős-Rényi hypergraphs. More precisely,  $k$ -clique counting on  $G(n, c, s)$  with  $k, c$  and  $s$  constant must take  $\tilde{\Omega}(n^{\omega \lceil k/3 \rceil})$  time when  $s = 2$  and  $\tilde{\Omega}(n^k)$  time when  $s \geq 3$ , unless there are faster worst-case algorithms. Furthermore, our reduction shows that it is ETH-hard to  $k$ -clique count in  $n^{o(k)}$  time on  $G(n, c, s)$  with  $k, c$  and  $s$  constant.
2. *Sparse Erdős-Rényi Hypergraphs.* Our reduction also applies with a different multiplicative slowdown and error tolerance to the sparse case of  $c = \Theta(n^{-\alpha})$ , where the fine-grained complexity of  $k$ -clique counting on  $G(n, c, s)$  is very different than on worst-case inputs. Our reduction implies fine-grained lower bounds of  $\tilde{\Omega}(n^{\omega \lceil k/3 \rceil - \alpha \binom{k}{2}})$  when  $s = 2$  and  $\tilde{\Omega}(n^{k - \alpha \binom{k}{s}})$  when  $s \geq 3$  for inputs drawn

---

<sup>2</sup>ETH asserts that 3-SAT in the worst-case takes at least  $2^{cn}$  time to solve for some constant  $c > 0$ .

from  $G(n, c, s)$ , unless there are faster worst-case algorithms. We remark that in the hypergraph case of  $s \geq 3$ , this lower bound matches the expected number of  $k$ -cliques up to  $\text{polylog}(n)$  factors.

Precise statements of our results can be found in Section 2.2. For simplicity, our results should be interpreted as applying to algorithms that succeed with probability  $1 - (\log n)^{-\omega(1)}$  in the dense case and  $1 - n^{-\omega(1)}$  in the sparse case, although our results apply in a more general context, as discussed in Section 2.2. We discuss the necessity of this error tolerance and the multiplicative slowdown in our worst-case to average-case reduction in Section 2.2. We also give a second worst-case to average-case reduction for computing the parity of the number of  $k$ -cliques which has weaker requirements on the error probability for the blackbox on  $G(n, c, s)$  in the dense case of  $c = 1/2$ .

We provide an overview of our multi-step worst-case to average-case reduction in Section 1.3. The steps are described in detail in Chapter 3.

**Algorithms for  $k$ -clique counting on  $G(n, c, s)$ .** We also analyze several natural algorithms for counting  $k$ -cliques in sparse Erdős-Rényi hypergraphs. These include an extension of the natural greedy algorithm mentioned previously from  $k$ -CLIQUE to counting  $k$ -cliques, a modification to this algorithm using the matrix multiplication step of [64] and an iterative algorithm achieving nearly identical guarantees. These algorithms count  $k$ -cliques in  $G(n, c, s)$  when  $c = \Theta(n^{-\alpha})$  in time:

- $\tilde{O}\left(n^{k+1-\alpha\binom{k}{s}}\right)$  if  $s \geq 3$  and  $k < \tau + 1$ ;
- $\tilde{O}\left(n^{\tau+2-\alpha\binom{\tau+1}{s}}\right)$  if  $s \geq 3$  and  $\tau + 1 \leq k \leq \kappa + 1$ ; and
- $\tilde{O}\left(n^{\omega\lceil k/3\rceil + \omega - \omega\alpha\binom{\lceil k/3\rceil}{2}}\right)$  if  $s = 2$  and  $k \leq \kappa + 1$ .

Here,  $\tau$  and  $\kappa$  are the largest positive integers satisfying that  $\alpha\binom{\tau}{s-1} < 1$  and  $\alpha\binom{\kappa}{s-1} < s$ . We restrict our attention to  $k$  with  $k \leq \kappa + 1$  since the probability that the largest clique in  $G$  has size  $\omega(G) > \kappa + 1$  is  $1/\text{poly}(n)$ . In the graph case of  $s = 2$ , these thresholds correspond to  $\alpha < \tau^{-1}$  and  $\alpha < 2\kappa^{-1} \leq \frac{2}{\kappa-1}$ . At  $k = \tau + 1$ , the

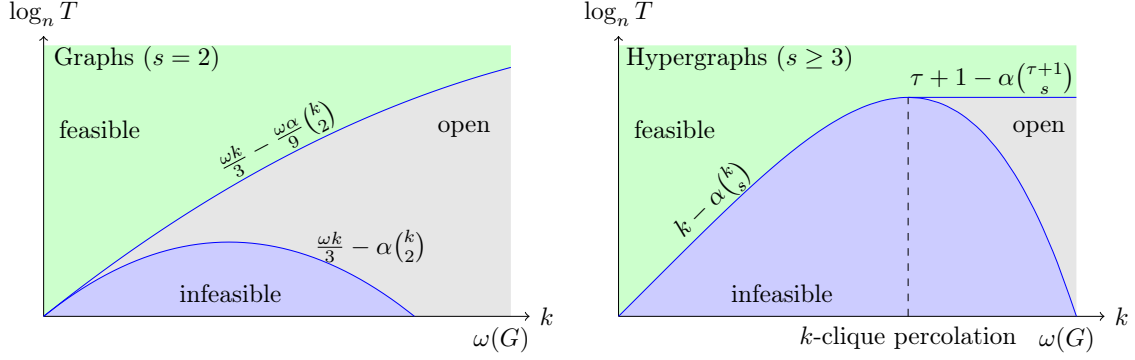


Figure 1-1: Comparison of our algorithms and average-case lower bounds for counting  $k$ -cliques in sparse Erdős-Rényi Hypergraphs  $G(n, c, s)$  with  $c = \Theta(n^{-\alpha})$ . Green denotes runtimes  $T$  feasible for each  $k$ , blue denotes  $T$  infeasible given that the best known worst-case algorithms are optimal and gray denotes  $T$  for which the complexity of counting  $k$ -cliques is open after this work. The left plot shows the graph case of  $s = 2$  and the right plot shows the hypergraph case of  $s \geq 3$ . For simplicity, all quantities shown are up to constant  $O_{k,\alpha}(1)$  additive error. ©2019 IEEE.

first threshold becomes  $\alpha < \frac{1}{k-1}$  which is exactly the  $k$ -clique percolation threshold [27, 65, 29]. Given a hypergraph  $G$ , define two  $k$ -cliques of  $G$  to be adjacent if they share  $(k-1)$  of their  $k$  vertices. This induces a hypergraph  $G_k$  on the set of  $k$ -cliques. For graphs  $G$  drawn from  $G(n, c)$ , [27] introduced the  $k$ -clique percolation threshold of  $c = \frac{1}{k-1} \cdot n^{-\frac{1}{k-1}}$ , above which a giant component emerges in  $G_k$ . This threshold and extensions were rigorously established in [15]. Following the same heuristic as in [27], our threshold  $\tau + 1$  is a natural extension of the  $k$ -clique percolation threshold to the hypergraph case of  $s \geq 3$ .

A comparison of our algorithmic guarantees and average-case lower bounds based on current best known worst-case algorithms for counting  $k$ -cliques is shown in Figure 1-1.

1. *Graph Case (s = 2)*. In the graph case, our lower and upper bounds have the same form and show that the exponent in the optimal running time is  $\frac{\omega k}{3} - C\alpha \binom{k}{2} + O_{k,\alpha}(1)$  where  $\frac{\omega}{9} \leq C \leq 1$  as long as  $k \leq \kappa + 1 = 2\alpha^{-1} + 1$ . As shown in Figure 1-1, our upper and lower bounds approach each other for  $k$  small relative to  $\kappa + 1$ .
2. *Hypergraph Case (s ≥ 3)*. In the hypergraph case of  $s \geq 3$ , the exponents in

our lower and upper bounds are nearly identical at  $k - \alpha \binom{k}{s} + O_{k,\alpha}(1)$  up to the  $k$ -clique percolation threshold. After this threshold, our lower bounds slowly deteriorate relative to our algorithms until they become trivial at the clique number of  $G$  by  $k = \kappa + 1$ .

Because we consider sparse Erdős-Rényi hypergraphs, for each  $n, k$ , and  $s$  we actually have an entire family of problems parametrized by the edge probability  $c$  and the behavior changes as a function of  $c$ ; this is the first worst-to-average-case hardness result we are aware of for which the complexity of the same problem over worst-case versus average-case inputs is completely different and can be sharply characterized over the whole range of  $c$  starting from the same assumption. It is surprising that our worst-case to average-case reduction techniques – which range from the self-reducibility of polynomials to random binary expansions – together yield tight lower bounds matching our algorithms in the hypergraph case. The fact that these lower bounds are tight exactly up the  $k$ -clique percolation threshold, a natural phase transition in the Erdős-Rényi model, is also unexpected a priori.

Two interesting problems left open after our work are to show average-case lower bounds with an improved constant  $C$  in the graph case and to show tight average-case lower bounds beyond the  $k$ -clique percolation threshold in the case  $s \geq 3$ . These and other open problems as well as some extensions of our methods are discussed in Chapter 6.

### 1.3 Overview of Reduction Techniques

For clarity of exposition, in this section we will restrict our discussion to the graph case  $s = 2$ , as well as the case of constant  $k$ .

A key step of our worst-case to average-case reduction uses the random self-reducibility of multivariate low-degree polynomials – i.e., evaluating a polynomial on any worst-case input can be efficiently reduced to evaluating it on several random inputs. This result follows from a line of work [59, 36, 41, 42] that provides a method to efficiently compute a polynomial  $P : \mathbb{F}^N \rightarrow \mathbb{F}$  of degree  $d \leq |\mathbb{F}|/20$  on any worst-

case input  $x \in \mathbb{F}^N$ , given an oracle  $\tilde{P} : \mathbb{F}^N \rightarrow \mathbb{F}$  that agrees with  $P$  on a  $\frac{1}{2} + \frac{1}{\text{poly}(N)}$  fraction of inputs. Thus, for any low-degree polynomial over a large enough finite field, evaluating the polynomial on a random element in the finite field is roughly as hard as evaluating the polynomial on any adversarially chosen input.

With the random self-reducibility of polynomials in mind, a natural approach is to express the number of  $k$ -cliques in a graph as a low-degree polynomial of the  $n \times n$  adjacency matrix  $A$

$$P(A) = \sum_{\substack{S \subseteq [n] \\ |S|=k}} \left( \prod_{i < j \in S} A_{ij} \right).$$

This polynomial has been used in a number of papers, including by Goldreich and Rothblum [44] to construct a distribution on dense graphs for which counting  $k$ -cliques is provably hard on average. However, the distribution they obtain is far from Erdős-Rényi and also their approach does not yield tight bounds for sparse graphs.

The significant obstacle that arises in applying the random self-reducibility of  $P$  is that one needs to work over a large enough finite field  $\mathbb{F}_p$ , so evaluating  $P$  on worst-case graph inputs in  $\{0, 1\}^{\binom{n}{2}}$  only reduces to evaluating  $P$  on uniformly random inputs in  $\mathbb{F}_p^{\binom{n}{2}}$ . In order to further reduce to evaluating  $P$  on graphs, given a random input  $A \in \mathbb{F}_p^{\binom{n}{2}}$  [44] uses several gadgets (including replacing vertices by independent sets and taking disjoint unions of graphs) in order to create a larger unweighted random graph  $A'$  whose  $k$ -clique count is equal to  $k! \cdot P(A) \pmod{p}$  for appropriate  $p$ . However, any nontrivial gadget-based reduction seems to have little hope of arriving at something close to the Erdős-Rényi distribution, because gadgets inherently create non-uniform structure.

We instead consider a different polynomial for graphs on  $nk$  vertices with  $nk \times nk$  adjacency matrix  $A$ ,

$$P'(A) = \sum_{v_1 \in [n]} \sum_{v_2 \in [2n] \setminus [n]} \cdots \sum_{v_k \in [kn] \setminus [(k-1)n]} \left( \prod_{1 \leq i < j \leq k} A_{v_i v_j} \right).$$

The polynomial  $P'$  correctly counts the number of  $k$ -cliques if  $A$  is  $k$ -partite with vertex  $k$ -partition  $[n] \sqcup ([2n] \setminus [n]) \sqcup \cdots \sqcup ([kn] \setminus [(k-1)n])$ . We first reduce clique-

counting in the worst case to computing  $P'$  in the worst case; this is a simple step, because it is a purely worst-case reduction. Next, we construct a recursive counting procedure that reduces evaluating  $P'$  on Erdős-Rényi graphs to counting  $k$ -cliques in Erdős-Rényi graphs. Therefore, it suffices to prove that if evaluating  $P'$  is hard in the worst case, then evaluating  $P'$  on Erdős-Rényi graphs is also hard.

Applying the Chinese Remainder theorem as well as the random self-reducibility of polynomials, computing  $P'$  on worst-case inputs in  $\{0, 1\}^{\binom{nk}{2}}$  reduces to computing  $P'$  on several uniformly random inputs in  $\mathbb{F}_p^{\binom{nk}{2}}$ , for several different primes  $p$  each on the order of  $\Theta(\log n)$ . The main question is: how can one evaluate  $P'$  on inputs  $X \sim \text{Unif}[\mathbb{F}_p^{\binom{nk}{2}}]$  using an algorithm that evaluates  $P'$  on  $G(n, c, 2)$  Erdős-Rényi graphs (i.e., inputs  $Y \sim \text{Ber}(c)^{\otimes \binom{nk}{2}}$ )?

To this end we introduce a method for converting finite field elements to binary expansions: an efficient rejection sampling procedure to find  $Y^{(0)}, \dots, Y^{(t)}$  (for  $t = \text{poly}(c^{-1}(1-c)^{-1} \log(n))$ ) such that each  $Y^{(i)}$  is close in total variation to  $\text{Ber}(c)^{\otimes \binom{nk}{2}}$ , and such that  $X = \sum_{i=0}^t 2^i Y^{(i)}$ . The correctness of the rejection sampling procedure is proved via a finite Fourier analytic method that bounds the total variation convergence of random biased binary expansions to the uniform distribution over residues in  $\mathbb{F}_p$ . This argument can be found in Chapter 4, and as discussed there the bounds we obtain are essentially optimal in their parameter dependence and this in turns yields near-optimal slowdown in the reduction. The technique appears likely to also be useful for other problems. For the unbiased case of  $c = 1/2$ , this binary expansions technique had previously appeared in [45].

Now we algebraically manipulate  $P'$  as follows:

$$\begin{aligned}
P'(X) &= \sum_{v_1 \in [n]} \sum_{v_2 \in [2n] \setminus [n]} \cdots \sum_{v_k \in [kn] \setminus [(k-1)n]} \prod_{1 \leq i < j \leq k} \left( \sum_{l \in \{0, \dots, t\}} 2^l \cdot Y_{v_i v_j}^{(l)} \right) \\
&= \sum_{f \in \{0, \dots, t\}^{\binom{k}{2}}} \left( \prod_{1 \leq i < j \leq k} 2^{f_{ij}} \right) \left( \sum_{v_1 \in [n]} \sum_{v_2 \in [2n] \setminus [n]} \cdots \sum_{v_k \in [kn] \setminus [(k-1)n]} \prod_{1 \leq i < j \leq k} Y_{v_i v_j}^{(f_{ij})} \right) \\
&= \sum_{f \in \{0, \dots, t\}^{\binom{k}{2}}} \left( \prod_{1 \leq i < j \leq k} 2^{f_{ij}} \right) P'(Y^{(f)}).
\end{aligned}$$

Here  $Y^{(f)}$  is the  $nk$ -vertex graph with entries given by  $Y_{ab}^{(f_{\bar{a}\bar{b}})}$  for  $1 \leq a < b \leq nk$ , where  $\bar{a} = \lceil a/n \rceil$  and  $\bar{b} = \lceil b/n \rceil$ . We thus reduce the computation of  $P'(X)$  to the computation of a weighted sum of  $\text{poly}(c^{-1}(1-c)^{-1} \log(n))^{\binom{k}{2}}$  different evaluations of  $P'$  at graphs close in total variation to Erdős-Rényi  $G(n, c, 2)$  graphs. This concludes our reduction. Notice that working with  $P'$  instead of  $P$  was necessary for the second equality.

We also give a different worst-case to average-case reduction for determining the parity of the number of  $k$ -cliques in Erdős-Rényi hypergraphs, as discussed in Section 2.2 and Chapter 3.

## 1.4 Notation and Preliminaries

A  $s$ -uniform hypergraph  $G = (V(G), E(G))$  consists of a vertex set  $V(G)$  and a hyperedge set  $E(G) \subseteq \binom{V(G)}{s}$ . A  $k$ -clique  $C$  in  $G$  is a subset of vertices  $C \subset V(G)$  of size  $|C| = k$  such that all of the possible hyperedges between the vertices are present in the hypergraph:  $\binom{C}{s} \subseteq E(G)$ . We write  $\text{cl}_k(G)$  to denote the set of  $k$ -cliques of the hypergraph  $G$ . One samples from the Erdős-Rényi distribution  $G(n, c, s)$  by independently including each of the  $\binom{n}{s}$  hyperedges with probability  $c$ .

We denote the law of a random variable  $X$  by  $\mathcal{L}(X)$ . We use  $T(A, n)$  to denote the worst-case run-time of an algorithm  $A$  on inputs of size parametrized by  $n$ . We work in the Word RAM model of computation, where the words have  $O(\log n)$  bits. All algorithms in this paper are randomized, and each (possibly biased) coin flip incurs constant computational cost.

## 1.5 Organization

Chapter 2 formally defines the various  $k$ -clique problems considered, and states the implications of the worst-case-to-average-case reduction proved in Chapter 3. Chapter 4 analyzes random binary expansions modulo a prime, which is needed in the worst-case-to-average-case reduction for sparse hypergraphs. Chapter 5 provides im-



proved algorithms for  $k$ -clique counting on sparse hypergraphs. And Chapter 6 concludes with extensions and open problems.



# Chapter 2

## Problem Formulations and Average-Case Lower Bounds

### 2.1 Clique Problems and Worst-Case Fine-Grained Conjectures

In this section, we formally define the problems we consider and the worst-case fine-grained complexity conjectures off of which our average-case lower bounds are based. We focus on the following computational problems.

**Definition 2.1.1.**  $\#(k, s)$ -CLIQUE denotes the problem of counting the number of  $k$ -cliques in an  $s$ -uniform hypergraph  $G$ .

**Definition 2.1.2.** PARITY- $(k, s)$ -CLIQUE denotes the problem of counting the number of  $k$ -cliques up to parity in an  $s$ -uniform hypergraph  $G$ .

**Definition 2.1.3.** DECIDE- $(k, s)$ -CLIQUE denotes the problem of deciding whether or not an  $s$ -uniform hypergraph  $G$  contains a  $k$ -clique.

Both  $\#(k, s)$ -CLIQUE and DECIDE- $(k, s)$ -CLIQUE are fundamental problems that have long been studied in computational complexity theory and are conjectured to be computationally hard. When  $k$  is allowed to be an unbounded input to the problem, DECIDE- $(k, s)$ -CLIQUE is known to be NP-complete [53] and  $\#(k, s)$ -CLIQUE

is known to be #P-complete [75]. In this work, we consider the fine-grained complexity of these problems, where  $k$  either can be viewed as a constant or a very slow-growing parameter compared to the number  $n$  of vertices of the hypergraph. In this context, PARITY- $(k, s)$ -CLIQUE can be interpreted as an intermediate problem between the other two clique problems that we consider. The reduction from PARITY- $(k, s)$ -CLIQUE to  $\#(k, s)$ -CLIQUE is immediate. And DECIDE- $(k, s)$ -CLIQUE reduces to PARITY- $(k, s)$ -CLIQUE with a  $O(k2^k)$ -multiplicative time overhead:

**Lemma 2.1.4.** *Given an algorithm  $A$  for PARITY- $(k, s)$ -CLIQUE that has error probability  $< 1/3$  on any  $s$ -uniform hypergraph  $G$ , there is an algorithm  $B$  that runs in time  $O(k2^k T(A, n))$  and solves DECIDE- $(k, s)$ -CLIQUE with error  $< 1/3$  on any  $s$ -uniform hypergraph  $G$ .*

*Proof.* Let  $\text{cl}_k(G)$  denote the set of  $k$ -cliques in hypergraph  $G = (V, E)$ . Consider the polynomial

$$P_G(x_V) = \sum_{S \in \text{cl}_k(G)} \prod_{v \in S} x_v \pmod{2},$$

over the finite field  $\mathbb{F}_2$ .

If  $G$  has a  $k$ -clique at vertices  $S \subset V$ , then  $P_G$  is nonzero, because  $P_G(1_S) = 1$ . If  $G$  has no  $k$ -clique, then  $P_G$  is zero. Therefore, deciding whether  $G$  has a  $k$ -clique reduces to testing whether or not  $P_G$  is identically zero.

$P_G$  is of degree at most  $k$ , so if  $P_G$  is nonzero on at least one input, we claim that it is nonzero on at least a  $2^{-k}$  fraction of inputs. The claim holds because if we evaluate  $P_G$  at all points  $a \in \{0, 1\}^m$ , the result is a non-zero Reed-Muller codeword in  $RM(k, m)$ , which has distance  $2^{m-k}$ , and block-length  $2^m$  [63].

We therefore evaluate  $P_G$  at  $c \cdot 2^k$  independent random inputs for some large enough  $c > 0$ , accept if any of the evaluations returns 1, and reject if all of the evaluations return 0. Each evaluation corresponds to calculating PARITY- $(k, s)$ -CLIQUE on a hypergraph  $G'$  formed from  $G$  by removing each vertex independently with probability  $1/2$ . As usual, we boost the error of  $A$  by running the algorithm  $O(k)$  times for each evaluation, and using the majority vote.  $\square$

When  $k$  is a constant, the trivial brute-force search algorithms for these problems are efficient in the sense that they take polynomial time. However, these algorithms do not remain efficient under the lens of fine-grained complexity since brute-force search requires  $\Theta(n^k)$  time, which can grow significantly as  $k$  grows. In the hypergraph case of  $s \geq 3$ , no algorithm taking time  $O(n^{k-\epsilon})$  on any of these problems is known, including for DECIDE- $(k, s)$ -CLIQUE [78]. In the graph case of  $s = 2$ , the fastest known algorithms take  $\Theta(n^{\omega^{\lceil k/3 \rceil}})$  time, where  $2 \leq \omega < 2.4$  is the fast matrix multiplication constant [48, 64]. Since this is the state of the art, one may conjecture that DECIDE- $(k, s)$ -CLIQUE and  $\#(k, s)$ -CLIQUE take  $n^{\Omega(k)}$  time in the worst case.

Supporting this conjecture, Razborov [68] proves that monotone circuits require  $\tilde{\Omega}(n^k)$  operations to solve DECIDE- $(k, 2)$ -CLIQUE in the case of constant  $k$ . Monotone circuit lower bounds are also known in the case when  $k = k(n)$  grows with  $n$  [2, 4]. In [30], DECIDE- $(k, 2)$ -CLIQUE is shown to be  $W[1]$ -hard. In other words, this shows that if DECIDE- $(k, 2)$ -CLIQUE is fixed-parameter tractable – admits an algorithm taking time  $f(k, s) \cdot \text{poly}(n)$  – then any algorithm in the parametrized complexity class  $W[1]$  is also fixed-parameter-tractable. This provides further evidence that DECIDE- $(k, 2)$ -CLIQUE is intractable for large  $k$ . Finally, [20] shows that solving DECIDE- $(k, 2)$ -CLIQUE in  $n^{o(k)}$  time is ETH-hard for constant  $k^1$ . We therefore conjecture that the  $k$ -clique problems take  $n^{\Omega(k)}$  time on worst-case inputs when  $k$  is constant, as formalized below.

**Conjecture 2.1.5** (Worst-case hardness of  $\#(k, s)$ -CLIQUE). *Let  $k$  be constant. Any randomized algorithm  $A$  for  $\#(k, s)$ -CLIQUE with error probability less than  $1/3$  takes time at least  $n^{\Omega(k)}$  in the worst case for hypergraphs on  $n$  vertices.*

**Conjecture 2.1.6** (Worst-case hardness of PARITY- $(k, s)$ -CLIQUE). *Let  $k$  be constant. Any randomized algorithm  $A$  for PARITY- $(k, s)$ -CLIQUE with error probability less than  $1/3$  takes time at least  $n^{\Omega(k)}$  in the worst case for hypergraphs on  $n$  vertices.*

---

<sup>1</sup>These hardness results also apply to DECIDE- $(k, s)$ -CLIQUE for  $s \geq 3$  since there is a reduction from DECIDE- $(k, 2)$ -CLIQUE to DECIDE- $(k, s)$ -CLIQUE in  $n^s$  time. The reduction proceeds by starting with a graph  $G$  and constructing an  $s$ -uniform hypergraph  $G'$  that contains a  $s$ -hyperedge for every  $s$ -clique in  $G$ . The  $k$ -cliques of  $G$  and  $G'$  are in bijection. This construction also reduces  $\#(k, 2)$ -CLIQUE to  $\#(k, s)$ -CLIQUE.

**Conjecture 2.1.7** (Worst-case hardness of DECIDE- $(k, s)$ -CLIQUE). *Let  $k$  be constant. Any randomized algorithm  $A$  for DECIDE- $(k, s)$ -CLIQUE with error probability less than  $1/3$  takes time at least  $n^{\Omega(k)}$  in the worst case for hypergraphs on  $n$  vertices.*

The conjectures are listed in order of increasing strength. Since Conjecture 2.1.7 is implied by ETH, they all follow from ETH. We also formulate a stronger version of the clique-counting hardness conjecture, which asserts that the current best known algorithms for  $k$ -clique counting are optimal.

**Conjecture 2.1.8** (Strong worst-case hardness of  $\#(k, s)$ -CLIQUE). *Let  $k$  be constant. Any randomized algorithm  $A$  for  $\#(k, s)$ -CLIQUE with error probability less than  $1/3$  takes time  $\tilde{\Omega}(n^{\omega\lceil k/3\rceil})$  in the worst case if  $s = 2$  and  $\tilde{\Omega}(n^k)$  in the worst case if  $s \geq 3$ .*

## 2.2 Average-Case Lower Bounds for Counting $k$ -Cliques in $G(n, c, s)$

Our first main result is a worst-case to average-case reduction solving either  $\#(k, s)$ -CLIQUE or PARITY- $(k, s)$ -CLIQUE on worst-case hypergraphs given a blackbox solving the problem on *most* Erdős-Rényi hypergraphs drawn from  $G(n, c, s)$ . We discuss this error tolerance over sampling Erdős-Rényi hypergraphs as well as the multiplicative overhead in our reduction below. These results show that solving the  $k$ -clique problems on Erdős-Rényi hypergraphs  $G(n, c, s)$  is as hard as solving them on worst-case hypergraphs, for certain choices of  $k, c$  and  $s$ . Therefore the worst-case hardness assumptions, Conjectures 2.1.5, 2.1.6 and 2.1.8, imply average-case hardness on Erdős-Rényi hypergraphs for  $\#(k, s)$ -CLIQUE and PARITY- $(k, s)$ -CLIQUE.

**Theorem 2.2.1** (Worst-case to average-case reduction for  $\#(k, s)$ -CLIQUE). *There is an absolute constant  $C > 0$  such that if we define*

$$\Upsilon_{\#}(n, c, s, k) \triangleq (C(c^{-1}(1-c)^{-1})(s \log k + s \log \log n)(\log n))^{\binom{k}{s}}$$

then the following statement holds. Let  $A$  be a randomized algorithm for  $\#(k, s)$ -CLIQUE with error probability less than  $1/\Upsilon_{\#}$  on hypergraphs drawn from  $G(n, c, s)$ . Then there exists an algorithm  $B$  for  $\#(k, s)$ -CLIQUE that has error probability less than  $1/3$  on any hypergraph, such that

$$T(B, n) \leq (\log n) \cdot \Upsilon_{\#} \cdot (T(A, nk) + (nk)^s)$$

For PARITY- $(k, s)$ -CLIQUE we also give an alternative reduction with an improved reduction time and error tolerance in the dense case when  $c = 1/2$ .

**Theorem 2.2.2** (Worst-case to average-case reduction for PARITY- $(k, s)$ -CLIQUE).

We have that:

1. There is an absolute constant  $C > 0$  such that if we define

$$\Upsilon_{P,1}(n, c, s, k) \triangleq \left( C(c^{-1}(1-c)^{-1})(s \log k) \left( s \log n + \binom{k}{s} \log \log \binom{k}{s} \right) \right)^{\binom{k}{s}}$$

then the following statement holds. Let  $A$  be a randomized algorithm for PARITY- $(k, s)$ -CLIQUE with error probability less than  $1/\Upsilon_{P,1}$  on hypergraphs drawn from  $G(n, c, s)$ . Then there exists an algorithm  $B$  for PARITY- $(k, s)$ -CLIQUE that has error probability less than  $1/3$  on any hypergraph, such that

$$T(B, n) \leq \Upsilon_{P,1} \cdot (T(A, nk) + (nk)^s)$$

2. There is an absolute constant  $C > 0$  such that if we define

$$\Upsilon_{P,2}(s, k) \triangleq (Cs \log k)^{\binom{k}{s}}$$

then the following statement holds. Let  $A$  be a randomized algorithm for PARITY- $(k, s)$ -CLIQUE with error probability less than  $1/\Upsilon_{P,2}$  on hypergraphs drawn from  $G(n, 1/2, s)$ . Then there exists an algorithm  $B$  for PARITY- $(k, s)$ -CLIQUE that

has error probability less than  $1/3$  on any hypergraph, such that

$$T(B, n) \leq \Upsilon_{P,2} \cdot (T(A, nk) + (nk)^s)$$

Our worst-case to average-case reductions yield the following fine-grained average-case lower bounds for  $k$ -clique counting and parity on Erdős-Rényi hypergraphs based on Conjectures 2.1.5 and 2.1.8. We separate these lower bounds into the two cases of dense and sparse Erdős-Rényi hypergraphs. We remark that, for all constants  $k$ , an error probability of less than  $(\log n)^{-\omega(1)}$  suffices in the dense case and error probability less than  $n^{-\omega(1)}$  suffices in the sparse case.

**Corollary 2.2.3** (Average-case hardness of  $\#(k, s)$ -CLIQUE on dense  $G(n, c, s)$ ). *If  $k, c, \epsilon > 0$  are constant, then we have that*

1. *Assuming Conjecture 2.1.5, then any algorithm  $A$  for  $\#(k, s)$ -CLIQUE that has error probability less than  $(\log n)^{-\binom{k}{s}-\epsilon}$  on Erdős-Rényi hypergraphs drawn from  $G(n, c, s)$  must have runtime at least  $T(A, n) \geq n^{\Omega(k)}$ .*
2. *Assuming Conjecture 2.1.8, then any algorithm  $A$  for  $\#(k, s)$ -CLIQUE that has error probability less than  $(\log n)^{-\binom{k}{s}-\epsilon}$  on Erdős-Rényi hypergraphs drawn from  $G(n, c, s)$  must have runtime at least  $T(A, n) \geq \tilde{\Omega}(n^{\omega\lceil k/3 \rceil})$  if  $s = 2$  and  $T(A, n) \geq \tilde{\Omega}(n^k)$  if  $s \geq 3$ .*

**Corollary 2.2.4** (Average-case hardness of  $\#(k, s)$ -CLIQUE on sparse  $G(n, c, s)$ ). *If  $k, \alpha, \epsilon > 0$  are constant and  $c = \Theta(n^{-\alpha})$ , then we have that*

1. *Assuming Conjecture 2.1.5, then any algorithm  $A$  for  $\#(k, s)$ -CLIQUE that has error probability less than  $n^{-\alpha\binom{k}{s}-\epsilon}$  on Erdős-Rényi hypergraphs drawn from  $G(n, c, s)$  must have runtime at least  $T(A, n) \geq n^{\Omega(k)}$ .*
2. *Assuming Conjecture 2.1.8, then any algorithm  $A$  for  $\#(k, s)$ -CLIQUE that has error probability less than  $n^{-\alpha\binom{k}{s}-\epsilon}$  on Erdős-Rényi hypergraphs drawn from  $G(n, c, s)$  must have runtime at least  $T(A, n) \geq \tilde{\Omega}(n^{\omega\lceil k/3 \rceil - \alpha\binom{k}{s}})$  if  $s = 2$  and  $T(A, n) \geq \tilde{\Omega}(n^{k - \alpha\binom{k}{s}})$  if  $s \geq 3$ .*



For PARITY- $(k, s)$ -CLIQUE, we consider here the implications of Theorem 2.2.2 only for  $c = 1/2$ , since this is the setting in which we obtain substantially different lower bounds than for  $\#(k, s)$ -CLIQUE. As shown, an error probability of  $o(1)$  on  $G(n, 1/2, s)$  hypergraphs suffices for our reduction to succeed.

**Corollary 2.2.5** (Average-case hardness of PARITY- $(k, s)$ -CLIQUE on  $G(n, 1/2, s)$ ). *Let  $k$  be constant. Assuming Conjecture 2.1.6, there is a small enough constant  $\epsilon \triangleq \epsilon(k, s)$  such that if any algorithm  $A$  for PARITY- $(k, s)$ -CLIQUE has error less than  $\epsilon$  on  $G(n, 1/2, s)$  then  $A$  must have runtime at least  $T(A, n) \geq n^{\Omega(k)}$ .*

We remark on one subtlety of our setup in the sparse case. Especially in our algorithms section, we generally restrict our attention to  $c = \Theta(n^{-\alpha})$  satisfying  $\alpha \leq s \binom{k}{s-1}^{-1}$ , which is necessary for the expected number of  $k$ -cliques in  $G(n, c, s)$  to not tend to zero. However, even when this expectation is decaying, the problem  $\#(k, s)$ -CLIQUE as we formulate it is still nontrivial. The simple algorithm that always outputs zero fails with a polynomially small probability that does not appear to meet the  $1/\Upsilon_{\#}$  requirement in our worst-case to average-case reduction. A simple analysis of this error probability can be found in Lemma 5.1.1. Note that even when  $\alpha > s \binom{k}{s-1}^{-1}$ , GREEDY-RANDOM-SAMPLING and its derivative algorithms in Chapter 5 still has guarantees and succeeds with probability  $1 - n^{-\omega(1)}$ . We now discuss the multiplicative overhead and error tolerance in our worst-case to average-case reduction for  $\#(k, s)$ -CLIQUE.

**Discussion of the Multiplicative Slowdown  $\Upsilon_{\#}$ .** In the sparse case of  $c = \Theta(n^{-\alpha})$ , our algorithmic upper bounds in Chapter 5 imply lower bounds on the necessary multiplicative overhead. In the hypergraph case of  $s \geq 3$  and below the  $k$ -clique percolation threshold, it must follow that the overhead is at least  $\Upsilon_{\#} = \tilde{\Omega}\left(n^{\alpha \binom{k}{s}}\right) = \tilde{\Omega}\left(c^{-\binom{k}{s}}\right)$ . Otherwise, our algorithms combined with our worst-case to average-case reduction would contradict Conjecture 2.1.8. Up to polylog( $n$ ) factors, this exactly matches the  $\Upsilon_{\#}$  from our reduction. In the graph case of  $s = 2$ , it similarly must follow that the overhead is at least  $\Upsilon_{\#} = \tilde{\Omega}\left(n^{\frac{\omega\alpha}{9} \binom{k}{s}}\right) = \tilde{\Omega}\left(c^{-\frac{\omega}{9} \binom{k}{s}}\right)$  to not contradict

Conjecture 2.1.8. This matches the  $\Upsilon_{\#}$  from our reduction up to a constant factor in the exponent.

**Discussion of the Error Tolerance**  $1/\Upsilon_{\#}$ . Notice that our worst-case to average-case reductions in Theorems 2.2.1 and 2.2.2 require that the error of the average-case blackbox on Erdős-Rényi hypergraphs go to zero as  $k$  goes to infinity. This error requirement can be seen to be unavoidable when  $k = \omega(\log n)$  in the dense Erdős-Rényi graph case of  $G(n, 1/2)$ . The expected number of  $k$ -cliques in  $G(n, 1/2)$  is  $\binom{n}{k} \cdot 2^{-\binom{k}{2}}$ , which is also an upper bound on the probability that  $G(n, 1/2)$  contains a  $k$ -clique by Markov's inequality.

If  $k \triangleq 3 \log_2 n$ , then the probability of a  $k$ -clique is less than  $n^k \cdot 2^{-k^2/2} = 2^{-k^2/6}$ . The algorithm that always outputs zero therefore achieves an average-case error of  $2^{-k^2/6}$  for  $\#(k, 2)$ -CLIQUE on  $G(n, 1/2)$ . However, this trivial algorithm is useless for solving  $\#(k, 2)$ -CLIQUE on worst-case inputs in a worst-case to average-case reduction. For this particular  $k = 3 \log_2 n$  regime, our  $\#(k, 2)$ -CLIQUE reduction requires average-case error on  $G(n, 1/2)$  less than  $1/\Upsilon_{\#} = 2^{-O(k^2 \log \log n)}$ . Our PARITY- $(k, 2)$ -CLIQUE reduction is more lenient, requiring error only less than  $2^{-O(k^2 \log \log \log n)}$  on  $G(n, 1/2)$ . Thus, the error bounds required by our reductions are quite close to the  $2^{-k^2/6}$  error bound that is absolutely necessary for any reduction in this regime. In the regime where  $k = O(1)$  is constant and on  $G(n, 1/2)$ , our PARITY- $(k, 2)$ -CLIQUE reduction only requires a small constant probability of error and our  $\#(k, 2)$ -CLIQUE reduction requires less than a  $1/\text{polylog}(n)$  probability of error. We leave it as an intriguing open problem whether the error tolerance of our reductions can be improved in this regime.

Finally, we remark that the error tolerance of the reduction must depend on  $c$ . By a union-bound on the  $k$ -subsets of vertices, the probability that a  $G(n, c)$  graph contains a  $k$ -clique is less than  $(n/c^{k/2})^k$ . For example, if  $c = 1/n$  then the probability that there exists a  $k$ -clique is less than  $n^{-\Omega(k^2)}$ . As a result, no worst-case to average-case reduction can tolerate average-case error more than  $n^{-O(k^2)}$  on  $G(n, 1/n)$  graphs. And therefore our reductions for  $\#(k, 2)$ -CLIQUE and for PARITY- $(k, 2)$ -CLIQUE are

close to optimal when  $c = 1/n$ , because our error tolerance in this case scales as  $n^{-O(k^2 \log \log n)}$ .



# Chapter 3

## Worst-Case to Average-Case

### Reduction for $G(n, c, s)$

In this section, we give our main worst-case to average-case reduction that transforms a blackbox solving  $\#(k, s)$ -CLIQUE on  $G(n, c, s)$  into a blackbox solving  $\#(k, s)$ -CLIQUE on a worst-case input hypergraph. This also yields a worst-case to average-case reduction for PARITY- $(k, s)$ -CLIQUE and proves Theorems 2.2.1 and 2.2.2. The reduction involves the following five main steps, the details of which are in Sections 3.1 to 3.5.

1. Reduce  $\#(k, s)$ -CLIQUE and PARITY- $(k, s)$ -CLIQUE on general worst-case hypergraphs to the worst-case problems with inputs that are  $k$ -partite hypergraphs with  $k$  parts of equal size.
2. Reduce the worst-case problem on  $k$ -partite hypergraphs to the problem of computing a low-degree polynomial  $P_{n,k,s}$  on  $N \triangleq N(n, k, s)$  variables over a small finite field  $\mathbb{F}$ .
3. Reduce the problem of computing  $P_{n,k,s}$  on worst-case inputs to computing  $P_{n,k,s}$  on random inputs in  $\mathbb{F}^N$ .
4. Reduce the problem of computing  $P_{n,k,s}$  on random inputs in  $\mathbb{F}^N$  to computing  $P_{n,k,s}$  on random inputs in  $\{0, 1\}^N$ . This corresponds to  $\#(k, s)$ -CLIQUE and

PARITY- $(k, s)$ -CLIQUE on  $k$ -partite Erdős-Rényi hypergraphs.

5. Reduce the average-case variants of  $\#(k, s)$ -CLIQUE and PARITY- $(k, s)$ -CLIQUE on  $k$ -partite Erdős-Rényi hypergraphs to non- $k$ -partite Erdős-Rényi hypergraphs.

These steps are combined in Section 3.6 to complete the proofs of Theorems 2.2.1 and 2.2.2. Before proceeding to our worst-case to average-case reduction, we establish some definitions and notation, and also give pseudocode for the counting reduction in Figure 3-1 – the parity reduction is similar.

The intermediate steps of our reduction crucially make use of  $k$ -partite hypergraphs with  $k$  parts of equal size, defined below.

**Definition 3.0.1** ( *$k$ -Partite Hypergraphs*). *Given a  $s$ -uniform hypergraph  $G$  on  $nk$  vertices with vertex set  $V(G) = [n] \times [k]$ , define the vertex labelling*

$$L : (i, j) \in [n] \times [k] \mapsto j \in [k]$$

*If for all  $e = \{u_1, \dots, u_s\} \in E(G)$ , the labels  $L(u_1), L(u_2), \dots, L(u_s)$  are distinct, then we say that  $G$  is  $k$ -partite with  $k$  parts of equal size  $n$ .*

In our reduction, it suffices to consider only  $k$ -partite hypergraphs with  $k$  parts of equal size. For ease of notation, our  $k$ -partite hypergraphs will always have  $nk$  vertices and vertex set  $[n] \times [k]$ . In particular, the edge set of a  $k$ -partite  $s$ -uniform hypergraph is an arbitrary subset of

$$E(G) \subseteq \{\{u_1, \dots, u_s\} \subset V(G) : L(u_1), \dots, L(u_s) \text{ are distinct}\}$$

Taking edge indicators yields that the  $k$ -partite hypergraphs on  $nk$  vertices we consider are in bijection with  $\{0, 1\}^N$ , where  $N \triangleq N(n, k, s) = \binom{k}{s} n^s$  is this size of this set of permitted hyperedges. Thus we will refer to elements  $x \in \{0, 1\}^N$  and  $k$ -partite  $s$ -uniform hypergraphs on  $nk$  vertices interchangeably. This definition also extends to Erdős-Rényi hypergraphs.

**Definition 3.0.2** (*k*-Partite Erdős-Rényi Hypergraphs). *The k-partite s-uniform Erdős-Rényi hypergraph  $G(nk, c, s, k)$  is a distribution over hypergraphs on  $nk$  vertices with vertex set  $V(G) = [n] \times [k]$ . A sample from  $G(nk, c, s, k)$  is obtained by independently including hyperedge each  $e = \{u_1, \dots, u_s\} \in E(G)$  with probability  $c$  for all  $e$  with  $L(u_1), L(u_2), \dots, L(u_s)$  distinct.*

Viewing the hypergraphs as elements of  $G(nk, c, s, k)$  as a distribution on  $\{0, 1\}^N$ , it follows that  $G(nk, c, s, k)$  corresponds to the product distribution  $\text{Ber}(c)^{\otimes N}$ .

### 3.1 Worst-Case Reduction to *k*-Partite Hypergraphs

In the following lemma, we prove that the worst-case complexity of  $\#(k, s)$ -CLIQUE and PARITY- $(k, s)$ -CLIQUE are nearly unaffected when we restrict the inputs to be worst-case *k*-partite hypergraphs. This step is important, because the special structure of *k*-partite hypergraphs will simplify future steps in our reduction.

**Lemma 3.1.1.** *Let  $A$  be an algorithm for  $\#(k, s)$ -CLIQUE, such that  $A$  has error probability less than  $1/3$  for any *k*-partite hypergraph  $G$  on  $nk$  vertices. Then, there is an algorithm  $B$  for  $\#(k, s)$ -CLIQUE with error probability less than  $1/3$  on any hypergraph  $G$  satisfying that  $T(B, n) \leq T(A, n) + O(k^s n^s)$ . Furthermore, the same result holds for PARITY- $(k, s)$ -CLIQUE in place of  $\#(k, s)$ -CLIQUE.*

*Proof.* Let  $G$  be an *s*-uniform hypergraph on  $n$  vertices. Construct the *s*-uniform hypergraph  $G'$  on the vertex set  $V(G') = [n] \times [k]$  with edge set

$$E(G') = \left\{ \{(v_1, t_1), (v_2, t_2), \dots, (v_s, t_s)\} : \{v_1, \dots, v_s\} \in E(G) \text{ and } \begin{array}{l} 1 \leq v_1 < v_2 < \dots < v_s \leq n \\ 1 \leq t_1 < t_2 < \dots < t_s \leq k \end{array} \right\}$$

The hypergraph  $G'$  can be constructed in  $O(k^s n^s)$  time. Note that  $G'$  is *k*-partite with the vertex partition  $L : (i, j) \in [n] \times [k] \mapsto j \in [k]$ . There is also a bijective correspondence between *k*-cliques in  $G'$  and *k*-cliques in  $G$  given by

$$\{v_1, v_2, \dots, v_k\} \mapsto \{(v_1, 1), (v_2, 2), \dots, (v_k, k)\}$$

where  $v_1 < v_2 < \dots < v_k$ . Thus, the  $k$ -partite  $s$ -uniform hypergraph  $G'$  on  $nk$  vertices has exactly the same number of  $k$ -cliques as  $G$ . It suffices to run  $A$  on  $G'$  and to return its output.  $\square$

A corollary to Lemma 3.1.1 is that if any worst-case hardness for  $\#(k, s)$ -CLIQUE and PARITY- $(k, s)$ -CLIQUE general  $s$ -uniform hypergraphs immediately transfers to the  $k$ -partite case. For instance, the lower bounds of Conjectures 2.1.5, 2.1.6, and 2.1.8 imply corresponding lower bounds in the  $k$ -partite case. Going forward in our worst-case to average-case reduction, we may restrict our attention to  $k$ -partite hypergraphs without loss of generality.

## 3.2 Counting $k$ -Cliques as a Low-Degree Polynomial

A key step in our worst-case to average-case reduction is to express the number of  $k$ -cliques as a low-degree polynomial in the adjacency matrix. As mentioned in the introduction, a similar step appears in the worst-case to average-case reduction of Goldreich and Rothblum [44].

In particular, the number of  $k$ -cliques of a  $k$ -partite hypergraph  $G$  with edge indicators  $x \in \{0, 1\}^N$  is a degree- $D$  polynomial  $P_{n,k,s} : \{0, 1\}^N \rightarrow \mathbb{Z}$  where  $D \triangleq D(k, s) = \binom{k}{s}$ . We identify the  $N$  coordinates of  $x \in \{0, 1\}^N$  with the  $s$ -subsets of  $[n] \times [k]$  with elements with all distinct labels, and for an  $s$ -vertex hyperedge  $S \subset V(G)$ , the variable  $x_S$  denotes the indicator variable that the hyperedge  $S$  is in the hypergraph  $x$ . The number of  $k$ -cliques in  $G$  is given by

$$P_{n,k,s}(x) = \sum_{\substack{\{u_1, \dots, u_k\} \subset V(G) \\ \forall i L(u_i) = i}} \prod_{\substack{S \subset [k] \\ |S| = s}} x_{u_S} \quad (3.1)$$

For any finite field  $\mathbb{F}$ , this equation defines  $P_{n,k,s}$  as a polynomial over that finite field. For clarity, we write this polynomial over  $\mathbb{F}$  as  $P_{n,k,s,\mathbb{F}} : \mathbb{F}^N \rightarrow \mathbb{F}$ . Observe that for any hypergraph  $x \in \{0, 1\}^N$ , we have that

$$P_{n,k,s,\mathbb{F}}(x) = P_{n,k,s}(x) \pmod{\text{char}(\mathbb{F})}$$



where  $\text{char}(\mathbb{F})$  is the characteristic of the finite field. We now reduce computing  $\#(k, s)$ -CLIQUE and PARITY- $(k, s)$ -CLIQUE on a  $k$ -partite hypergraph  $x \in \{0, 1\}^N$  to computing  $P_{n,k,s,\mathbb{F}}(x)$  for appropriate finite fields  $\mathbb{F}$ . This is formalized in the following two propositions.

**Proposition 3.2.1.** *Let  $x \in \{0, 1\}^N$  denote a  $s$ -uniform hypergraph that is  $k$ -partite with vertex labelling  $L$ . Let  $p_1, p_2, \dots, p_t$  be  $t$  distinct primes, such that  $\prod_i p_i > n^k$ . Then, solving  $\#(k, s)$ -CLIQUE reduces to computing  $P_{n,k,s,\mathbb{F}_{p_i}}(x)$  for all  $i \in [t]$ , plus  $O(k \log n)$  additive computational overhead. Moreover, computing  $P_{n,k,s,\mathbb{F}_{p_i}}(x)$  for all  $i \in [t]$  reduces to computing  $\#(k, s)$ -CLIQUE, plus  $O(tk \log n)$  computational overhead.*

*Proof.* Note that  $P_{n,k,s}(x) \leq n^k$  since there are at most  $n^k$  cliques in the hypergraph. So the claim follows from the Chinese Remainder Theorem and the fact that for any  $i \in [t]$ , it holds that  $P_{n,k,s,\mathbb{F}_{p_i}}(x) \equiv P_{n,k,s}(x) \pmod{p_i}$ .  $\square$

**Proposition 3.2.2.** *Let  $\mathbb{F}$  be a finite field of characteristic 2. Let  $x \in \{0, 1\}^N$  be a  $s$ -uniform hypergraph that is  $k$ -partite with vertex labelling  $L$ . Then solving PARITY- $(k, s)$ -CLIQUE for  $x$  is equivalent to computing  $P_{n,k,s,\mathbb{F}}(x)$ .*

*Proof.* This is immediate from  $P_{n,k,s,\mathbb{F}}(x) \equiv P_{n,k,s}(x) \pmod{\text{char}(\mathbb{F})}$ .  $\square$

### 3.3 Random Self-Reducibility: Reducing to Random Inputs in $\mathbb{F}^N$

Expressing the number and parity of cliques as low-degree polynomials allows us to perform a key step in the reduction: because polynomials over finite fields are random self-reducible, we can reduce computing  $P_{n,k,s,\mathbb{F}}$  on worst-case inputs to computing  $P_{n,k,s,\mathbb{F}}$  on several uniformly random inputs in  $\mathbb{F}^N$ .

The following well-known lemma states the random self-reducibility of low-degree polynomials. The lemma first appeared in [42]. We follow the proof of [8] in order to present the lemma with explicit guarantees on the running time of the reduction.

**Lemma 3.3.1** (cf. Theorem 4 of [42]). *Let  $\mathbb{F}$  be a finite field with  $|\mathbb{F}| = q$  elements. Let  $N, D > 0$ . Suppose  $9 < D < q/12$ . Let  $f : \mathbb{F}^N \rightarrow \mathbb{F}$  be a polynomial of degree at most  $D$ . If there is an algorithm  $A$  running in time  $T(A, N)$  such that*

$$\mathbb{P}_{x \sim \text{Unif}[\mathbb{F}^N]}[A(x) = f(x)] > 2/3,$$

*then there is an algorithm  $B$  running in time  $O((N + D)D^2 \log^2 q + T(A, N) \cdot D)$  such that for any  $x \in \mathbb{F}^N$ , it holds that  $\mathbb{P}[B(x) = f(x)] > 2/3$ .*

*Proof.* Our proof of the lemma is based off of the proof that appears in [8]. The only difference is that in [8], the lemma is stated only for finite fields whose size is a prime.

Suppose we wish to calculate  $f(x)$  for  $x \in \mathbb{F}^N$ . In order to do this, choose  $y_1, y_2 \stackrel{i.i.d}{\sim} \text{Unif}[\mathbb{F}^N]$ , and define the polynomial  $g(t) = x + ty_1 + t^2 y_2$  where  $t \in \mathbb{F}$ . We evaluate  $A(g(t))$  at  $m$  different values  $t_1, \dots, t_m \in \mathbb{F}$ . This takes  $O(mND \log^2 q + m \cdot T(A, N))$  time.

Suppose that we have the guarantee that at most  $(m - 2D)/2$  of these evaluations are incorrect. Then, since  $f(g(t))$  is a univariate polynomial of degree at most  $2D$ , we may use Berlekamp-Welch to recover  $f(g(0)) = A(x)$  in  $O(m^3)$  arithmetic operations over  $\mathbb{F}$ , each of which takes  $O(\log^2 q)$  time.

Since  $g(t_i)$  and  $g(t_j)$  are pairwise independent and uniform in  $\mathbb{F}^N$  for any distinct  $t_i, t_j \neq 0$ , by the second-moment method, with probability  $> 2/3$ , at most  $(m - 2D)/2$  evaluations of  $A(g(t))$  will be incorrect if we take  $m = 12D$ .  $\square$

Lemma 3.3.1 implies that if we can efficiently compute  $P_{n,k,s,\mathbb{F}}$  on at least a  $2/3$  fraction of randomly chosen inputs in  $\mathbb{F}^N$ , then we can efficiently compute the polynomial  $P_{n,k,s,\mathbb{F}}$  over a worst-case input in  $\mathbb{F}^N$ .

### 3.4 Reduction to Evaluating the Polynomial on $G(nk, c, s, k)$

So far, we have reduced worst-case clique-counting over unweighted hypergraphs to the average-case problem of computing  $P_{n,k,s,\mathbb{F}}$  over  $k$ -partite hypergraphs with random edge weights in  $\mathbb{F}$ . It remains to reduce from computing  $P_{n,k,s,\mathbb{F}}$  on inputs

$x \sim \text{Unif}[\mathbb{F}^N]$  to random hypergraphs, which correspond to  $x \sim \text{Unif}[\{0, 1\}^N]$ . Since  $\{0, 1\}^N$  is an exponentially small subset of  $\mathbb{F}^N$  if  $|\mathbb{F}| > 2$ , the random weighted and unweighted hypergraph problems are very different. In this section, we carry out this reduction using two different arguments for PARITY- $(k, s)$ -CLIQUE and  $\#(k, s)$ -CLIQUE. The latter reduction is based on the total variation convergence of random binary expansion modulo  $p$  to  $\text{Unif}[\mathbb{F}_p]$  and related algorithmic corollaries from Chapter 4.

We first present the reduction that will be applied in the case of PARITY- $(k, s)$ -CLIQUE. Given a map  $a : \binom{[k]}{s} \rightarrow \{0, 1, \dots, t-1\}$ , let  $a^* : [N] \rightarrow \{0, 1, \dots, t-1\}$  denote the map induced by the labels  $L : V(G) \rightarrow [k]$  of the vertices, when the indices of  $[N]$  are identified with the possible  $k$ -partite hyperedges of  $G$ . Explicitly, if  $v$  is a bijection between  $[N]$  and the set of possible  $k$ -partite hyperedges of  $G$  under the labelling  $L$ , then define  $a^*(i) = a(L(v(i)))$  for all  $i \in [N]$ . Recall that  $D = \binom{k}{s}$  is the degree of  $P_{n,k,s}$ . The following lemma will be used only for the PARITY- $(k, s)$ -CLIQUE case:

**Lemma 3.4.1.** *Let  $p$  be prime and  $t \geq 1$ . Suppose  $A$  is an algorithm that computes  $P_{n,k,s,\mathbb{F}_p}(y)$  with error probability less than  $\delta \triangleq \delta(n)$  for  $y \sim \text{Unif}[\mathbb{F}_p^N]$  in time  $T(A, n)$ . Then there is an algorithm  $B$  that computes  $P_{n,k,s,\mathbb{F}_{p^t}}(x)$  with error probability less than  $t^D \cdot \delta$  for  $x \sim \text{Unif}[\mathbb{F}_{p^t}^N]$  in time  $T(B, n) = O(Nt^4(\log p)^3 + t^D \cdot T(A, n))$ .*

*Proof.* We give a reduction computing  $P_{n,k,s,\mathbb{F}_{p^t}}(x)$  where  $x \sim \text{Unif}[\mathbb{F}_{p^t}^N]$  given black-box access to  $A$ . Let  $\beta$  be such that  $\beta, \beta^p, \beta^{p^2}, \dots, \beta^{p^{t-1}} \in \mathbb{F}_{p^t}$  forms a normal basis for  $\mathbb{F}_{p^t}$  over  $\mathbb{F}_p$ . Now for each  $i \in [N]$ , compute the basis expansion

$$x_i = x_i^{(0)}\beta + x_i^{(1)}\beta^p + \dots + x_i^{(t-1)}\beta^{p^{t-1}}.$$

One can find a generator for a normal basis  $\beta \in \mathbb{F}_{p^t}$  in time  $O((t^2 + \log p)(t \log p)^2)$  by Bach et al. [7]. Computing  $x^{(0)}, \dots, x^{(t-1)}$  then takes time  $O(Nt^3(\log p)^3)$  because  $N$  applications of Gaussian elimination each take at most  $O(t^3)$  operations over  $\mathbb{F}_p$ .<sup>1</sup> Note that since  $x$  is uniformly distributed and  $\beta, \beta^p, \dots, \beta^{p^{t-1}}$  form a basis, it follows

---

<sup>1</sup>For a good survey on normal bases, we recommend [40].

that  $x^{(0)}, x^{(1)}, \dots, x^{(t-1)}$  are distributed i.i.d according to  $\text{Unif}[\mathbb{F}_p^N]$ . For any map  $b : [N] \rightarrow \{0, 1, \dots, t-1\}$  define  $x^{(b)} \in \mathbb{F}_p^N$  as  $x_i^{(b)} = x_i^{(b(i))}$  for all  $i \in [N]$ . Observe that for any fixed map  $b$ , the vector  $x^{(b)}$  is uniform in  $\mathbb{F}_p^N$ . We now expand and redistribute the terms of  $P_{n,k,s,\mathbb{F}_p^t}$  as follows.

$$\begin{aligned}
P_{n,k,s,\mathbb{F}_p^t}(x) &= \sum_{\substack{\{u_1, \dots, u_k\} \subset V(G) \\ \forall i L(u_i)=i}} \prod_{S \in \binom{[k]}{s}} x_{u_S} \\
&= \sum_{\substack{\{u_1, \dots, u_k\} \subset V(G) \\ \forall i L(u_i)=i}} \prod_{S \in \binom{[k]}{s}} \left( \sum_{i=0}^{t-1} x_{u_S}^{(i)} \beta^{p^i} \right) \\
&= \sum_{a: \binom{[k]}{s} \rightarrow \{0, \dots, t-1\}} \left( \sum_{\substack{\{u_1, \dots, u_k\} \subset V(G) \\ \forall i L(u_i)=i}} \prod_{S \in \binom{[k]}{s}} \left( x_{u_S}^{(a(S))} \beta^{p^{a(S)}} \right) \right) \\
&= \sum_{a: \binom{[k]}{s} \rightarrow \{0, \dots, t-1\}} \left( \prod_{S \in \binom{[k]}{s}} \beta^{p^{a(S)}} \right) \left( \sum_{\substack{\{u_1, \dots, u_k\} \subset V(G) \\ \forall i L(u_i)=i}} \prod_{S \in \binom{[k]}{s}} x_{u_S}^{(a(S))} \right) \\
&= \sum_{a: \binom{[k]}{s} \rightarrow \{0, \dots, t-1\}} \left( \prod_{S \in \binom{[k]}{s}} \beta^{p^{a(S)}} \right) P_{n,k,s,\mathbb{F}_p}(x^{(a^*)})
\end{aligned}$$

As observed above, it holds that  $x^{(a^*)} \sim \text{Unif}[\mathbb{F}_p^N]$  for each  $a$ . Thus, computing  $P_{n,k,s,\mathbb{F}_p}(x)$  reduces to evaluating  $P_{n,k,s,\mathbb{F}_p}$  on  $t^D$  uniformly random inputs on in  $\mathbb{F}_p^N$  and outputting a weighted sum of the evaluations. The desired bound on the error probability follows from a union bound.  $\square$

We now give the reduction to evaluating  $P_{n,k,s}$  on random hypergraphs drawn from  $G(nk, c, s, k)$  in the case of  $\#(k, s)$ -CLIQUE.

**Lemma 3.4.2.** *Let  $p$  be prime and let  $c = c(n), \gamma = \gamma(n) \in (0, 1)$ . Suppose that  $A$  is an algorithm that computes  $P_{n,k,s,\mathbb{F}_p}(y)$  with error probability less than  $\delta \triangleq \delta(n)$  when  $y \in \{0, 1\}^N$  is drawn from  $G(nk, c, s, k)$ . Then, for some  $t = O(c^{-1}(1-c)^{-1} \log(Np/\gamma) \log p)$ , there is an algorithm  $B$  that evaluates  $P_{n,k,s,\mathbb{F}_p}(x)$  with error probability at most  $\gamma + t^D \cdot \delta$  when  $x \sim \text{Unif}[\mathbb{F}_p^N]$  in time  $T(B, n) =$*

$O(Npt \log(Np/\gamma) + t^D \cdot T(A, n))$ .

*Proof.* We give a reduction computing  $P_{n,k,s,\mathbb{F}_p}(x)$  where  $x \sim \text{Unif}[\mathbb{F}_p^N]$  given blackbox access to  $A$ . We first handle the case in which  $p > 2$ . For each  $j \in [N]$ , apply the algorithm from Lemma 4.0.3 to sample  $x_j^{(0)}, x_j^{(1)}, \dots, x_j^{(t-1)} \in \{0, 1\}$  satisfying

$$d_{\text{TV}}\left(\mathcal{L}(x_j^{(0)}, \dots, x_j^{(t-1)}), \text{Ber}(c)^{\otimes t}\right) \leq \epsilon \triangleq \gamma/N \quad \text{and} \quad \sum_{i=0}^{t-1} 2^i x_j^{(i)} \equiv x_j \pmod{p}$$

By Lemmas 4.0.2 and 4.0.3, we may choose  $t = O(c^{-1}(1-c)^{-1} \log(Np/\gamma) \log p)$  and this sampling can be carried out in  $O(Npt \log(Np/\gamma))$  time. By the total variation bound, for each  $j$  we may couple  $(x_j^{(0)}, \dots, x_j^{(t-1)})$  with  $(Z_j^{(0)}, \dots, Z_j^{(t-1)}) \sim \text{Ber}(c)^{\otimes t}$ , so that  $\mathbb{P}[x_j^{(i)} = Z_j^{(i)} \forall i, j] \geq 1 - \gamma$ . Moreover, we have  $x_j^{(i)} \perp x_l^{(k)}$  whenever  $j \neq l$ , so we may choose the  $Z_j^{(i)}$  so that  $Z_j^{(i)} \perp Z_l^{(k)}$  whenever  $j \neq l$ .

As in the proof of Lemma 3.4.1, given any map  $b : [N] \rightarrow \{0, \dots, t-1\}$ , we define  $Z^{(b)} \in \{0, 1\}^N$  by  $Z_j^{(b)} = Z_j^{(b(j))}$ , for all  $j \in [N]$ . We also note that for any fixed  $b$ , the entries  $Z_1^{(b)}, \dots, Z_N^{(b)}$  are independent and distributed as  $\text{Ber}(c)$ . Therefore,

$$Z^{(b)} \sim G(nk, c, s, k)$$

Now compute the following quantity, similarly to the calculations in Lemma 3.4.1:

$$\begin{aligned} \tilde{P}_{n,k,s,\mathbb{F}_p}(Z) &\triangleq \sum_{\substack{\{u_1, \dots, u_k\} \subset V(G) \\ \forall i \ L(u_i)=i}} \prod_{S \in \binom{[k]}{s}} \left( \sum_{i=0}^{t-1} 2^i \cdot Z_{u_S}^{(i)} \right) \\ &= \sum_{a: \binom{[k]}{s} \rightarrow \{0, \dots, t-1\}} \left( \sum_{\substack{\{u_1, \dots, u_k\} \subset V(G) \\ \forall i \ L(u_i)=i}} \prod_{S \in \binom{[k]}{s}} (2^{a(S)} \cdot Z_{u_S}^{(a(S))}) \right) \\ &= \sum_{a: \binom{[k]}{s} \rightarrow \{0, \dots, t-1\}} \left( \prod_{S \in \binom{[k]}{s}} 2^{a(S)} \right) \left( \sum_{\substack{\{u_1, \dots, u_k\} \subset V(G) \\ \forall i \ L(u_i)=i}} \prod_{S \in \binom{[k]}{s}} Z_{u_S}^{(a(S))} \right) \\ &= \sum_{a: \binom{[k]}{s} \rightarrow \{0, \dots, t-1\}} \left( \prod_{S \in \binom{[k]}{s}} 2^{a(S)} \right) P_{n,k,s,\mathbb{F}_p}(Z^{(a^*)}). \end{aligned}$$

We may use algorithm  $A$  to evaluate the  $t^D$  values of  $P_{n,k,s,\mathbb{F}_p}(Z^{(a^*)})$ , with probability  $< t^D \cdot \delta$  of any error (by a union bound). Computing  $\tilde{P}_{n,k,s,\mathbb{F}_p}(Z)$  reduces to computing a weighted sum over the  $t^D$  evaluations. Conditioned on the event that  $x_j^{(i)} = Z_j^{(i)} \forall i, j$ , then  $P_{n,k,s,\mathbb{F}_p}(x) = \tilde{P}_{n,k,s,\mathbb{F}_p}(Z)$ , because

$$\begin{aligned} P_{n,k,s,\mathbb{F}_p}(x) &= \sum_{\substack{\{u_1, \dots, u_k\} \subset V(G) \\ \forall i L(u_i) = i}} \prod_{S \in \binom{[k]}{s}} x_{u_S} \\ &= \sum_{\substack{\{u_1, \dots, u_k\} \subset V(G) \\ \forall i L(u_i) = i}} \prod_{S \in \binom{[k]}{s}} \left( \sum_{i=0}^{t-1} 2^i \cdot x_{u_S}^{(i)} \right) \\ &= \sum_{\substack{\{u_1, \dots, u_k\} \subset V(G) \\ \forall i L(u_i) = i}} \prod_{S \in \binom{[k]}{s}} \left( \sum_{i=0}^{t-1} 2^i \cdot Z_{u_S}^{(i)} \right) = \tilde{P}_{n,k,s,\mathbb{F}_p}(Z). \end{aligned}$$

Since  $\mathbb{P}[x_j^{(i)} = Z_j^{(i)} \forall i, j] \geq 1 - t \cdot \gamma$ , by a union bound with the error in calculation we have computed  $P_{n,k,s,\mathbb{F}_p}(x)$  with probability of error  $\leq \gamma + t^D \cdot \delta$ . The claim follows for the case  $p > 2$ .

If  $p = 2$ , then the proof is almost identical, except that since  $2 \equiv 0 \pmod{2}$ , we may no longer use the result on random binary expansions of Lemma 4.0.3. In this case, for each  $j \in [N]$  we sample  $x_j^{(0)}, \dots, x_j^{(t-1)} \in \{0, 1\}^N$  such that each  $d_{\text{TV}}(\mathcal{L}(x_j^{(0)}, \dots, x_j^{(t-1)}), \text{Ber}(c)^{\otimes t}) \leq \epsilon \triangleq \gamma/N$ , and so that

$$\sum_{i=0}^{t-1} x_j^{(i)} = x_j \pmod{p}.$$

By Lemma 4.0.4, we may choose  $t = O(c^{-1}(1-c)^{-1} \log(N/\gamma))$ , and we may sample in time  $O(Nt \log(N/\gamma))$ . Again, we couple the  $x_j^{(i)}$  variables with variables  $Z_j^{(i)} \sim \text{Ber}(c)$  such that the event  $E$  that  $x_j^{(i)} = Z_j^{(i)}$  for all  $i, j$  has probability  $\mathbb{P}[E] \geq 1 - \gamma$  and such that  $Z_j^{(i)}$  is independent of  $Z_l^{(k)}$  whenever  $j \neq l$ . By a similar, and simpler, calculation to the one for the case  $p > 2$ , we have that  $\tilde{P}_{n,k,s,\mathbb{F}_2}(Z) = P_{n,k,s,\mathbb{F}_2}(x)$  conditioned on

$E$ , where

$$\tilde{P}_{n,k,s,\mathbb{F}_2}(Z) \triangleq \sum_{a: \binom{[k]}{s} \rightarrow \{0, \dots, t-1\}} P_{n,k,s,\mathbb{F}_2}(Z^{(a^*)}).$$

This can be calculated using the algorithm  $A$  similarly to the  $p > 2$  case, because each  $Z^{(a^*)}$  is distributed as  $G(nk, c, s, k)$ .  $\square$

### 3.5 Reduction to Counting $k$ -Cliques in $G(n, c, s)$

So far, we have reduced PARITY- $(k, s)$ -CLIQUE and  $\#(k, s)$ -CLIQUE for worst-case input hypergraphs to average-case inputs drawn from the  $k$ -partite Erdős-Rényi distribution  $G(nk, c, s, k)$ . We now carry out the final step of the reduction, showing that PARITY- $(k, s)$ -CLIQUE and  $\#(k, s)$ -CLIQUE on inputs drawn from  $G(nk, c, s, k)$  reduce to inputs drawn from the non- $k$ -partite Erdős-Rényi distribution  $G(n, c, s)$ . Recall that a hypergraph  $G$  drawn from  $G(nk, c, s, k)$  has vertex set  $V(G) = [n] \times [k]$  and vertex partition given by the labels  $L : (i, j) \in [n] \times [k] \mapsto j \in [k]$ .

**Lemma 3.5.1.** *Let  $\delta = \delta(n) \in (0, 1)$  be a non-increasing function of  $n$  and let  $c = c(n) \in (0, 1)$ . Suppose that  $A$  is a randomized algorithm for  $\#(k, s)$ -CLIQUE such that for any  $n$ ,  $A$  has error probability less than  $\delta(n)$  on hypergraphs drawn from  $G(n, c, s)$  in  $T(A, n)$  time. Then there exists an algorithm  $B$  solving  $\#(k, s)$ -CLIQUE that has error probability less than  $2^k \cdot \delta(n)$  on hypergraphs drawn from  $G(nk, c, s, k)$  and that runs in  $T(B, n) = O(2^k \cdot T(A, nk) + k^s n^s + k2^k)$  time.*

*Proof.* It suffices to count the number of  $k$ -cliques in  $G \sim G(nk, c, s, k)$  given blackbox access to  $A$ . Construct the hypergraph  $H$  over the same vertex set  $V(H) = [n] \times [k]$  by adding each edge  $e = \{v_1, v_2, \dots, v_s\} \in \binom{[n] \times [k]}{s}$  such that  $|\{L(v_1), \dots, L(v_s)\}| < s$  independently with probability  $c$ . In other words, independently add each edge to  $G$  containing two vertices from the same part of  $G$ . It follows that  $H$  is distributed according to  $G(nk, c, s)$ . More generally, for every  $S \subset [k]$ ,  $H_S$  is distributed according to  $G(|L^{-1}(S)|, c, s)$  where  $H_S$  is the restriction of  $H$  to the vertices  $L^{-1}(S) \subset V(H)$  with labels in  $S$ . Note that  $H$  can be constructed in  $O(k^s n^s)$  time.

Now observe that for each  $S \neq \emptyset$ , it holds that  $n \leq |L^{-1}(S)| \leq nk$  and the algorithm  $A$  succeeds on each  $H_S$  with probability at least  $1 - \delta(n)$ . By a union bound, we may compute the number of  $k$ -cliques  $|\text{cl}_k(H_S)|$  in  $H_S$  for all  $S \subset [k]$  with error probability less than  $2^k \cdot \delta(n)$ . Note that this can be done in  $O(2^k \cdot T(A, nk))$  time. From these counts  $|\text{cl}_k(H_S)|$ , we now to inductively compute

$$t_d \triangleq |\{S \in \text{cl}_k(H) : |L(S)| = d\}|$$

for each  $d \in [k]$ . Note that  $t_0 = 0$  in the base case  $d = 0$ . Given  $t_0, t_1, \dots, t_d$ , the next count  $t_{d+1}$  can be expressed by inclusion-exclusion as

$$\begin{aligned} t_{d+1} &= \sum_{T \subset [k], |T|=d+1} |\{S \in \text{cl}_k(H) : L(S) = T\}| \\ &= \sum_{T \subset [k], |T|=d+1} \left( |\text{cl}_k(H_T)| - \sum_{i=0}^d \sum_{U \subset T, |U|=i} |\{S \in \text{cl}_k(H) : L(S) = U\}| \right) \\ &= \left( \sum_{T \subset [k], |T|=d+1} |\text{cl}_k(H_T)| \right) - \sum_{i=0}^d \binom{k-i}{d+1-i} |\{S \in \text{cl}_k(H) : |L(S)| = i\}| \\ &= \sum_{T \subset [k], |T|=d+1} |\text{cl}_k(H_T)| - \sum_{i=0}^d \binom{k-i}{d+1-i} t_i \end{aligned}$$

After  $O(k2^k)$  operations, this recursion yields the number of  $k$ -cliques  $t_k = |\{S \in \text{cl}_k(H) : |L(S)| = k\}| = |\text{cl}_k(G)|$  in the original  $k$ -partite hypergraph  $G$ , as desired.  $\square$

Repeating the same proof over  $\mathbb{F}_2$  yields an analogue of Lemma 3.5.1 for PARITY- $(k, s)$ -CLIQUE, as stated below.

**Lemma 3.5.2.** *Lemma 3.5.1 holds when  $\#(k, s)$ -CLIQUE is replaced by PARITY- $(k, s)$ -CLIQUE.*

### 3.6 Proofs of Theorems 2.2.1 and 2.2.2

We now combine Steps 1-5 formally in order to prove Theorems 2.2.1 and 2.2.2.



*Proof of Theorem 2.2.1.* Our goal is to construct an algorithm  $B$  that solves  $\#(k, s)$ -CLIQUE with error probability  $< 1/3$  on any  $s$ -uniform hypergraph  $x$ .

We are given an algorithm  $A$  that solves  $\#(k, s)$ -CLIQUE with probability of error  $< 1/\Upsilon_{\#}$  on hypergraphs drawn from  $G(n, c, s)$ .

We will construct the following intermediate algorithms in our reduction:

- Algorithm  $A_0$  that solves  $\#(k, s)$ -CLIQUE with error probability  $< 1/3$  for any worst-case  $k$ -partite hypergraph.
- Algorithm  $A_1(x, p)$  that computes  $P_{n,k,s,\mathbb{F}_p}(x)$  for any  $x \in \mathbb{F}_p^N$  and for any prime  $p$  such that  $12\binom{k}{s} < p < 10 \log n^k$ , with worst-case error probability  $< 1/3$ .
- Algorithm  $A_2(y, p)$  for primes  $12\binom{k}{s} < p < 10 \log n^k$  that computes  $P_{n,k,s,\mathbb{F}_p}(y)$  on inputs  $y \sim \text{Unif}[\mathbb{F}_p^N]$  with error probability  $< 1/3$ .
- Algorithm  $A_3(z)$  that computes  $P_{n,k,s}(z)$  on inputs  $z \sim G(nk, c, s, k)$  with error probability  $< \delta$ . (The required value of  $\delta$  will be determined later on.)

We construct algorithm  $B$  from  $A_0$ ,  $A_0$  from  $A_1$ ,  $A_2$  from  $A_3$ , and  $A_3$  from  $A$ .

1. *Reduce to computing  $\#(k, s)$ -CLIQUE for  $k$ -partite hypergraphs.* We use Lemma 3.1.1 to construct  $B$  from  $A_0$ , such that  $B$  runs in time

$$T(B, n) = T(A_0, n) + O((nk)^s).$$

2. *Reduce to computing  $P_{n,k,s,\mathbb{F}_p}$  on worst-case inputs.* We use Proposition 3.2.1 to construct  $A_0$  from  $A_1$  such that  $A_0$  runs in time

$$T(A_0, n) \leq O(T(A_1, n) \cdot \log n^k + (\log n^k)^2).$$

The algorithm  $A_0$  starts by using a sieve to find the first  $T$  primes  $12\binom{k}{s} < p_1 < \dots < p_T$  such that  $\prod_{i=1}^T p_i > n^k$ . Notice that  $p_T \leq 10 \log n^k$ , so this step takes time  $O((\log n^k)^2)$ . Then, given a  $k$ -partite hypergraph  $x \in \{0, 1\}^N$ , the algorithm

$A_0$  computes  $P_{n,k,s}(x)$  by computing  $P_{n,k,s,\mathbb{F}_{p_i}}(x)$  for all  $p_i$ , boosting the error of  $A_1$  by repetition and majority vote. Since  $T = O((\log n^k)/(\log \log n^k))$ , we only need to repeat  $O(\log \log n^k)$  times per prime; this yields a total slowdown factor of  $O(\log n^k)$ . Once we have computed  $P_{n,k,s}(x)$ , we recall that it is equal to the number of  $k$ -cliques in  $x$ .

3. *Reduce to computing  $P_{n,k,s,\mathbb{F}_p}$  on random inputs in  $\mathbb{F}_p^N$ .* We use Lemma 3.3.1 to construct  $A_1$  from  $A_2$  such that  $A_2$  runs in time

$$T(A_1, n) = O((N+D)D^2 \log^2 p + D \cdot T(A_2, n)) = O(n^s \binom{k}{s}^2 \log^2 \log n^k + \binom{k}{s} \cdot T(A_2, n)).$$

4. *Reduce to computing  $P_{n,k,s}$  on random inputs in  $\{0, 1\}^N$*  We use Lemma 3.4.2 to construct  $A_2$  from  $A_3$  such that  $A_2$  runs in time

$$T(A_2, n) = O(Npt \log(Np) + t \binom{k}{s} \cdot T(A_3, n)),$$

for some  $t = O(c^{-1}(1-c)^{-1}s(\log n)(\log p))$ . For this step, we require the error probability  $\delta$  of algorithm  $A_3(z)$  on inputs  $z \sim G(nk, c, s, k)$  to be at most  $1/(4t^D) = 1/(4t \binom{k}{s})$ .

5. *Reduce to computing  $\#(k, s)$ -CLIQUE for  $G(n, c, s)$  hypergraphs* We use Lemma 3.5.1 to construct  $A_3$  from  $A$  such that  $A_3$  runs in time

$$T(A_3, n) = O((nk)^s + k2^k + 2^k \cdot T(A, nk)),$$

and such that  $A_3$  has error probability at most  $\delta < 2^k/\Upsilon_{\#}$ .

As in the theorem statement, let  $\Upsilon_{\#}(n, c, s, k) \triangleq (C(c^{-1}(1-c)^{-1}s(\log n)(\log k + \log \log n)) \binom{k}{s})$ , where  $C > 0$  is a large constant to be determined. If we take  $C$  large enough, then  $4t \binom{k}{s} \cdot 2^k \leq \Upsilon_{\#}$ . In this case, the error  $\delta$  of  $A_3$  will be at most  $1/(4t \binom{k}{s})$ ,

which is what we needed for the fourth step. Putting the runtime bounds together,

$$\begin{aligned} T(B, n) &= O((nk)^s + (\log nk)^2 + (\log nk) \cdot (n^s tk \binom{k}{s})^2 (\log n)^2 + \binom{k}{s} \cdot (4t)^{\binom{k}{s}} \cdot (T(A, nk) + (nk)^s)) \\ &= O(n^s k^3 \binom{k}{s}^2 (c^{-1}(1-c)^{-1})(\log k + \log \log n) \log^4 n + (\log n) \cdot \Upsilon_{\#} \cdot (T(A, nk) + (nk)^s)), \end{aligned}$$

if we choose  $C > 0$  large enough. Hence,

$$T(B, n) = O((\log n) \cdot \Upsilon_{\#} \cdot (T(A, nk) + (nk)^s)),$$

as  $\binom{k}{s} \geq 3$  without loss of generality.  $\square$

*Proof of Theorem 2.2.2.* The proof of item 1 of Theorem 2.2.2 is analogous to the proof of Theorem 2.2.1, except that it does not use the Chinese remainder theorem. Moreover, special care is needed in order to ensure that the field  $\mathbb{F}$  over which we compute the polynomial  $P_{n,k,s,\mathbb{F}}$  in the intermediate steps is large enough that we may use the random self-reducibility of polynomials.

Our goal is to construct an algorithm  $B$  that solves PARITY- $(k, s)$ -CLIQUE with error probability  $< 1/3$  on any  $s$ -uniform hypergraph  $x$ .

We are given an algorithm  $A$  that solves PARITY- $(k, s)$ -CLIQUE with probability of error  $< 1/\Upsilon_{P,1}$  on hypergraphs drawn from  $G(n, c, s)$ .

We will construct the following intermediate algorithms in our reduction:

- Algorithm  $A_0$  that solves PARITY- $(k, s)$ -CLIQUE with error probability  $< 1/3$  for any worst-case  $k$ -partite hypergraph.
- Algorithm  $A_1(w)$  that computes  $P_{n,k,s,\mathbb{F}_{2^\kappa}}(w)$  on inputs  $w \sim \text{Unif}[\mathbb{F}_{2^\kappa}^N]$  for  $\kappa = \lceil \log_2(12 \binom{k}{s}) \rceil$ , with error probability  $< 1/3$ .
- Algorithm  $A_2(y)$  that computes  $P_{n,k,s,\mathbb{F}_2}(y)$  on inputs  $y \sim \text{Unif}[\mathbb{F}_2^N]$  with error probability  $< \delta_2$ . (The required value of  $\delta_2$  will be determined later on.)
- Algorithm  $A_3(z)$  that computes  $P_{n,k,s,\mathbb{F}_2}(z)$  on inputs  $z \sim G(nk, c, s, k)$  with error probability  $< \delta_3$ . (The required value of  $\delta_3$  will be determined later on.)

We construct algorithm  $B$  from  $A_0$ ,  $A_0$  from  $A_1$ ,  $A_2$  from  $A_3$ , and  $A_3$  from  $A$ .

1. *Reduce to computing PARITY- $(k, s)$ -CLIQUE for  $k$ -partite hypergraphs.* We use Lemma 3.1.1 to construct  $B$  from  $A_0$ , such that  $B$  runs in time

$$T(B, n) = T(A_0, n) + O((nk)^s).$$

2. *Reduce to computing  $P_{n,k,s,\mathbb{F}_{2^\kappa}}$  on random inputs in  $\mathbb{F}_{2^\kappa}^N$ .* Note that by Proposition 3.2 if we can compute  $P_{n,k,s,\mathbb{F}_{2^\kappa}}$  for worst-case inputs, then we can solve PARITY- $(k, s)$ -CLIQUE. We use Lemma 3.3.1 to construct  $A_0$  from  $A_1$  such that  $A_0$  runs in time

$$T(A_0, n) = O(\kappa^2(N + D)D^2 + D \cdot T(A_1, n)) = O(n^s \binom{k}{s}^2 \log^2 \kappa + \binom{k}{s} \cdot T(A_1, n))$$

3. *Reduce to computing  $P_{n,k,s,\mathbb{F}_2}$  on random inputs in  $\mathbb{F}_2^N$ .* We use Lemma 3.4.1 to construct  $A_1$  from  $A_2$  such that  $A_1$  runs in time

$$T(A_1, n) \leq O(N\kappa^4 + \kappa \binom{k}{s} \cdot T(A_2, n)),$$

and has error probability at most  $\delta_2 \cdot \kappa \binom{k}{s}$  on random inputs  $w \sim \text{Unif}[\mathbb{F}_{2^\kappa}^N]$ . Thus,  $A_2$  must have error probability at most  $\delta_2 < 1/(3\kappa \binom{k}{s})$  on random inputs in  $y \sim \text{Unif}[\mathbb{F}_2^N]$  for this step of the reduction to work.

4. *Reduce to computing  $P_{n,k,s,\mathbb{F}_2}$  on random inputs in  $\{0, 1\}^N$*  We use Lemma 3.4.2 to construct  $A_2$  from  $A_3$  such that  $A_2$  runs in time

$$T(A_2, n) = O(Nt \log(N/\gamma) + t \binom{k}{s} \cdot T(A_3, n)),$$

for some  $t = O(c^{-1}(1 - c)^{-1}(s \log(n) + \log(1/\gamma)))$ . The error probability of  $A_2$  on random inputs  $z \sim G(nk, c, s, k)$  will be at most  $\delta_2 < \delta_3 \cdot t \binom{k}{s} + \gamma$ . Since we require error probability at most  $\delta_2 \leq 1/(3\kappa \binom{k}{s})$  of algorithm  $A_2(z)$  on inputs  $z \sim G(nk, c, s, k)$ , we set  $\gamma = 1/(10\kappa \binom{k}{s})$  and require  $\delta_3 \leq 1/(10(t\kappa) \binom{k}{s})$ , which is sufficient. For this choice

of  $\gamma$ , we have  $t = O(c^{-1}(1-c)^{-1}(s \log(n) + \binom{k}{s} \log(s \log k)))$ .

5. *Reduce to computing  $\#(k, s)$ -CLIQUE for  $G(n, c, s)$  hypergraphs* We use Lemma 3.5.2 to construct  $A_3$  from  $A$  such that  $A_3$  runs in time

$$T(A_3, n) = O((nk)^s + k2^k + 2^k \cdot T(A, nk)),$$

and such that  $A_3$  has error probability at most  $\delta_3 < 2^k/\Upsilon_{P,1}$ .

As in the theorem statement, let

$$\Upsilon_{P,1}(n, c, s, k) \triangleq \left( C(c^{-1}(1-c)^{-1})s(\log k)(s \log n + \binom{k}{s}(\log \log \binom{k}{s})) \right)^{\binom{k}{s}}$$

for some large enough constant  $C$ .

If we take  $C$  large enough, then  $(\kappa t)^{\binom{k}{s}} \leq \frac{1}{10} \cdot 2^{-k} \cdot \Upsilon_{P,1}$ , as desired. In this case, the error of  $A_0$  on uniformly random inputs will be at most  $1/3$ , which is what we needed. Putting the runtime bounds together,

$$\begin{aligned} T(B, n) &= O(n^s \binom{k}{s}^2 \log^2 \kappa + n^s t \log(n^s \kappa^{\binom{k}{s}}) + n^s \binom{k}{s} \kappa^4 + \binom{k}{s} \cdot (4\kappa t)^{\binom{k}{s}} \cdot (T(A, nk) + (nk)^s)) \\ &= O(n^s (t \binom{k}{s})^2 \log^2 s \log k + \binom{k}{s} \kappa^4) + \Upsilon_{P,1} \cdot (T(A, nk) + (nk)^s), \end{aligned}$$

if we choose  $C > 0$  large enough. Since  $\binom{k}{s} \geq 3$  without loss of generality,

$$T(B, n) = O(\Upsilon_{P,1} \cdot (T(A, nk) + (nk)^s)).$$

For item 2 of the theorem, we restrict the inputs to come from  $G(n, 1/2, s)$ , and we achieve a better error tolerance because algorithm  $A_3$  is the same as  $A_2$ . This means that we may skip step 4 of the proof of item 1. In particular, we only need  $\delta_3 = \delta_2 \leq 1/(3\kappa^{\binom{k}{s}})$ . So algorithm  $A$  only needs to have error  $< 1/\Upsilon_{P,2}$ , for  $\Upsilon_{P,2}(k, s) \triangleq (Cs \log k)^{\binom{k}{s}}$ . It is not hard to see that, skipping step 4, the algorithm  $B$  that we construct takes time  $T(B, n) = O(\Upsilon_{P,2} \cdot (T(A, nk) + (nk)^s))$ .  $\square$

**Algorithm** TO-ER- $\#(G, k, A, c)$ 

*Inputs:*  $s$ -uniform hypergraph  $G$  with vertex set  $[n]$ , parameters  $k, c$ , algorithm  $A$  for  $\#(k, s)$ -CLIQUE on Erdős-Rényi hypergraphs with density  $c$ .

1. Construct an  $s$ -uniform hypergraph  $G'$  on vertex set  $[n] \times [k]$  by defining

$$E(G') = \left\{ \{(v_1, t_1), (v_2, t_2), \dots, (v_s, t_s)\} : \{v_1, \dots, v_s\} \in E(G) \text{ and } \begin{array}{l} 1 \leq v_1 < v_2 < \dots < v_s \leq n \\ 1 \leq t_1 < t_2 < \dots < t_s \leq k \end{array} \right\}.$$

Since  $G'$  is  $k$ -partite, view it as an indicator vector of edges  $G' \in \{0, 1\}^N$  for  $N := N(n, k, s) = \binom{k}{s} n^s$ .

2. Find the first  $T$  primes  $12 \binom{k}{s} < p_1 < \dots < p_T$  such that  $\prod_{i=1}^T p_i > n^k$ .
3. Define  $L : (a, b) \in [n] \times [k] \mapsto b \in [k]$ , and

$$P_{n,k,s}(x) = \sum_{\substack{\{u_1, \dots, u_k\} \in V(G') \\ L(u_i) = i \ \forall i}} \prod_{\substack{S \subseteq [k] \\ |S|=s}} x_{u_S}$$

For each  $1 \leq t \leq T$ , compute  $P_{n,k,s}(G') \pmod{p_t}$ , as follows:

- (1) Use the procedure of [42] in order to reduce the computation of  $P_{n,k,s}(G') \pmod{p_t}$  to the computation of  $P_{n,k,s}$  on  $M = 12 \binom{k}{s}$  distinct inputs  $x_1, \dots, x_M \sim \text{Unif}[\mathbb{F}_{p_t}^N]$ .
- (2) For each  $1 \leq m \leq M$ , compute  $P_{n,k,s}(x_m) \pmod{p_t}$  as follows:
  - (i) Use the rejection sampling procedure of Lemma 4.0.3 in order to sample  $(Y^{(0)}, \dots, Y^{(B)})$  close to  $(\text{Ber}(c)^{\otimes N})^{\otimes B}$  in total variation distance, such that  $x_m \equiv \sum_{b=0}^B 2^b \cdot Y^{(b)} \pmod{p_t}$ . It suffices to take  $B = \Theta(c^{-1}(1 - c)^{-1} s (\log n) (\log p_t))$ .
  - (ii) For each function  $a : \binom{[k]}{s} \rightarrow \{0, \dots, B\}$ , define  $Y_S^{(a)} = Y^{a(L(S))}$  for all  $S \in [N] \subset \binom{[n]}{s}$ . Note that for each  $a$ , the corresponding  $Y^{(a)}$  is approximately distributed as  $\text{Ber}(c)^{\otimes N}$ . Use algorithm  $A$  and the recursive counting procedure of Lemma 3.5.1 in order to compute  $P_{n,k,s}(Y^{(a)})$  for each  $a$ .
  - (iii) Set  $P_{n,k,s}(G') \leftarrow \sum_{a: \binom{[k]}{s} \rightarrow \{0, \dots, B\}} 2^{|a|_1} \cdot P_{n,k,s}(Y^{(a)})$ .
4. Since  $0 \leq P_{n,k,s}(G') \leq n^k$ , use Chinese remaindering and the computations of  $P_{n,k,s}(G') \pmod{p_i}$  in order to calculate and output  $P_{n,k,s}(G')$ .

Figure 3-1: Reduction TO-ER- $\#$  for showing computational lower bounds for average-case  $\#(k, s)$ -CLIQUE on Erdős-Rényi  $G(n, c, s)$  hypergraphs based on the worst-case hardness of  $\#(k, s)$ -CLIQUE. ©2019 IEEE.

# Chapter 4

## Random Binary Expansions Modulo $p$

In this section, we consider the distributions of random binary expansions of the form

$$Z_t \cdot 2^t + Z_{t-1} \cdot 2^{t-1} + \cdots + Z_0 \pmod{p}$$

for some prime  $p$  and independent, possibly biased, Bernoulli random variables  $Z_i \in \{0, 1\}$ . We show that for  $t$  polylogarithmic in  $p$ , these distributions become close to uniformly distributed over  $\mathbb{F}_p$ , more or less regardless of the biases of the  $Z_i$ . This is then used to go in the other direction, producing approximately independent Bernoulli variables that are the binary expansion of a number with a given residue. The special case of this argument in which the Bernoulli variables are unbiased has already appeared in an earlier work by Goldreich and Rothblum [45]. In that case, the proof of correctness is much simpler, because the Fourier-analytic tools used below can be avoided.

Our argument uses finite Fourier analysis on  $\mathbb{F}_p$ . Given a function  $f : \mathbb{F}_p \rightarrow \mathbb{R}$ , define its Fourier transform to be  $\hat{f} : \mathbb{F}_p \rightarrow \mathbb{C}$ , where  $\hat{f}(t) = \sum_{x=0}^{p-1} f(x)\omega^{tx}$  and  $\omega = e^{2\pi i/p}$ . In this section, we endow  $\mathbb{F}_p$  with the total ordering of  $\{0, 1, \dots, p-1\}$  as elements of  $\mathbb{Z}$ . Given a set  $S$ , let  $2S = \{2s : s \in S\}$ . We begin with a simple claim showing that sufficiently long geometric progressions with ratio 2 in  $\mathbb{F}_p$  contain a middle residue modulo  $p$ .

**Claim 4.0.1.** *Suppose that  $a_1, \dots, a_k \in \mathbb{F}_p$  is a sequence with  $a_1 \neq 0$  and  $a_{i+1} = 2a_i$*

for each  $1 \leq i \leq k-1$ . Then if  $k \geq 1 + \log_2(p/3)$ , there is some  $j$  with  $\frac{p}{3} \leq a_j \leq \frac{2p}{3}$ .

*Proof.* Let  $S = \{x \in \mathbb{F}_p : x < p/3\}$  and  $T = \{x \in \mathbb{F}_p : x > 2p/3\}$ . Observe that  $2S \cap T = \emptyset$  and  $S \cap 2T = \emptyset$ , which implies that no two consecutive  $a_i$  can be in  $S$  and  $T$ . Therefore if  $(a_1, a_2, \dots, a_k)$  contains elements of both  $S$  and  $T$ , there must be some  $j$  with  $a_j \in (S \cup T)^C$  and the claim follows. It thus suffices to show that  $(a_1, a_2, \dots, a_k)$  cannot be entirely contained in one of  $S$  or  $T$ . First consider the case that it is contained in  $S$ . Define the sequence  $(a'_1, a'_2, \dots, a'_k)$  of integers by  $a'_{i+1} = 2a'_i$  for each  $1 \leq i \leq k-1$  and  $a'_1 \in [1, p/3)$  is such that  $a'_1 \equiv a_1 \pmod{p}$ . It follows that  $a'_i \equiv a_i \pmod{p}$  for each  $i$  and  $a'_k \geq 2^{k-1} \geq p/3$ . Now consider the smallest  $j$  with  $a'_j > p/3$ . Then  $p/3 \geq a'_{j-1} = a'_j/2$  by the minimality of  $j$ , and  $p/3 \leq a_j \leq 2p/3$  which is a contradiction. If the sequence is contained in  $T$ , then  $(-a_1, -a_2, \dots, -a_k)$  is contained in  $S$  and applying the same argument to this sequence proves the claim.  $\square$

We now prove the main lemma of this section bounding the total variation between the distribution of random binary expansions modulo  $p$  and the uniform distribution.

**Lemma 4.0.2.** *Let  $p > 2$  be prime. Suppose that  $c \leq q_0, q_1, \dots, q_t \leq 1 - c$  for some  $c \in (0, 1/2]$  and  $\epsilon > 0$ . Then there is an absolute constant  $K > 0$  such that if  $t \geq K \cdot c^{-1}(1 - c)^{-1} \log(p/\epsilon^2) \log p$  and  $Z_i \sim \text{Ber}(q_i)$  are independent, then the distribution of  $S = \sum_{i=0}^t Z_i \cdot 2^i \pmod{p}$  is within  $\epsilon$  total variation distance of the uniform distribution on  $\mathbb{F}_p$ .*

*Proof.* Let  $f : \mathbb{F}_p \rightarrow \mathbb{R}$  be the probability mass function of  $\sum_{i=0}^t 2^i Z_i \pmod{p}$ . By definition, we have that

$$f(x) = \sum_{z \in \{0,1\}^{t+1}} \left( \prod_{i=0}^t q_i^{z_i} (1 - q_i)^{1-z_i} \right) \mathbf{1} \left\{ \sum_{i=0}^t z_i \cdot 2^i \equiv x \pmod{p} \right\}$$

This definition and factoring yields that  $\hat{f}(s)$  is given by

$$\hat{f}(s) = \sum_{x=0}^{p-1} f(x) \omega^{sx} = \prod_{i=0}^t \left( 1 - q_i + q_i \cdot \omega^{2^i \cdot s} \right)$$



Note that the constant function  $\mathbf{1}$  has Fourier transform  $p \cdot \mathbf{1}_{\{s=0\}}$ . By Cauchy-Schwarz and Parseval's theorem, we have that

$$\begin{aligned} 4 \cdot d_{\text{TV}}(\mathcal{L}(S), \text{Unif}[\mathbb{F}_p])^2 &= \|f - p^{-1} \cdot \mathbf{1}\|_1^2 \leq p \cdot \|f - p^{-1} \cdot \mathbf{1}\|_2^2 = \|\hat{f} - \mathbf{1}_{\{s=0\}}\|_2^2 \\ &= \sum_{s \neq 0} \prod_{i=0}^t \left| 1 - q_i + q_i \cdot \omega^{2^i \cdot s} \right|^2 \end{aligned}$$

Note that  $|1 - q + q \cdot \omega^a| \leq 1$  by the triangle inequality for all  $a \in \mathbb{F}_p$  and  $q \in (0, 1)$ . Furthermore, if  $a \in \mathbb{F}_p$  is such that  $p/3 \leq a \leq 2p/3$  and  $q \in [c, 1 - c]$ , then we have that

$$\begin{aligned} |1 - q + q \cdot \omega^a|^2 &= (1 - q)^2 + q^2 + 2q(1 - q) \cos(2\pi a/p) \\ &= 1 - 2q(1 - q) (1 - \cos(2\pi a/p)) \\ &\leq 1 - 2c(1 - c) (1 - \cos(4\pi/3)) \\ &= 1 - 3c(1 - c) \end{aligned}$$

since  $\cos(x)$  is maximized at the endpoints on the interval  $x \in [2\pi/3, 4\pi/3]$  and  $q(1 - q)$  is minimized at the endpoints on the interval  $[c, 1 - c]$ . Now suppose that  $t$  is such that

$$t \geq \left\lceil \frac{\log(4\epsilon^2/p)}{\log(1 - 3c(1 - c))} \right\rceil \cdot \lceil 1 + \log_2(p/3) \rceil = \Theta(c^{-1}(1 - c)^{-1} \log(p/\epsilon^2) \log p)$$

Fix some  $s \in \mathbb{F}_p$  with  $s \neq 0$ . By Claim 4.0.1, any  $\lceil 1 + \log_2(p/3) \rceil$  consecutive terms of the sequence  $s, 2s, \dots, 2^t s \in \mathbb{F}_p$  contain an element between  $p/3$  and  $2p/3$ . Therefore this sequence contains at least  $m = \left\lceil \frac{\log(4\epsilon^2/p)}{\log(1 - 3c(1 - c))} \right\rceil$  such terms, which implies that

$$\prod_{i=0}^t \left| 1 - q_i + q_i \cdot \omega^{2^i \cdot s} \right|^2 \leq (1 - 3c(1 - c))^m \leq \frac{4\epsilon^2}{p}$$

by the inequalities above. Since this holds for each  $s \neq 0$ , it now follows that

$$4 \cdot d_{\text{TV}}(\mathcal{L}(S), \text{Unif}[\mathbb{F}_p])^2 \leq \sum_{s \neq 0} \prod_{i=0}^t \left| 1 - q_i + q_i \cdot \omega^{2^i \cdot s} \right|^2 < 4\epsilon^2$$

and thus  $d_{\text{TV}}(\mathcal{L}(S), \text{Unif}[\mathbb{F}_p]) < \epsilon$ , proving the lemma.  $\square$

We now briefly discuss the tightness of the bounds on  $t$  in the lemma above and how the case of  $c = 1/2$  differs from  $c \neq 1/2$ . Note that if  $q_i = 1/2$  for each  $i$ , then  $\sum_{i=0}^t Z_i \cdot 2^i$  is uniformly distributed on  $\{0, 1, \dots, 2^{t+1} - 1\}$ . It follows that

$$d_{\text{TV}}(\mathcal{L}(S), \text{Unif}[\mathbb{F}_p]) = \sum_{x \in \mathbb{F}_p} |p^{-1} - \mathbb{P}[S = x]|_+ = \frac{a(p-a)}{2^{t+1}p} \leq \frac{p}{2^{t+1}}$$

if  $0 \leq a \leq p-1$  is such that  $2^{t+1} \equiv a \pmod{p}$ . Therefore  $S$  is within total variation of  $1/\text{poly}(p)$  of  $\text{Unif}[\mathbb{F}_p]$  if  $t = \Omega(\log p)$ . However, note that for  $c$  constant and  $\epsilon = 1/\text{poly}(p)$ , our lemma requires that  $t = \Omega(\log^2 p)$ . This raises the question: is the additional factor of  $\log p$  necessary or an artefact of our analysis? We answer this question with an example suggesting that the extra  $\log p$  factor is in fact necessary and that the case  $c = 1/2$  is special.

Suppose that  $p$  is a Mersenne prime with  $p = 2^r - 1$  for some prime  $r$  and for simplicity, take  $q_i = 1/3$  for each  $i$ . Observe by the triangle inequality that

$$\left| \hat{f}(1) \right| = \left| \sum_{x \in \mathbb{F}_p} (f(x) - p^{-1}) \cdot \omega^x \right| \leq \|f - p^{-1} \cdot \mathbf{1}\|_1 = 2 \cdot d_{\text{TV}}(\mathcal{L}(S), \text{Unif}[\mathbb{F}_p])$$

Now suppose that  $t = ar - 1$  for some positive integer  $a$ . As shown in the lemma, we have

$$\left| \hat{f}(1) \right|^2 = \prod_{i=0}^t \left| \frac{2}{3} + \frac{1}{3} \cdot \omega^{2^i} \right|^2 = \left[ \prod_{i=0}^{r-1} \left( \frac{5}{9} + \frac{4}{9} \cdot \cos \left( \frac{2\pi}{p} \cdot 2^i \right) \right) \right]^a$$

where the second equality is due to the fact that the sequence  $2^i$  has period  $r$  modulo

$p$ . Now observe that since  $\frac{5}{9} + \frac{4}{9} \cdot \cos(x) \geq e^{-x^2}$ , we have that

$$\prod_{i=0}^{r-1} \left( \frac{5}{9} + \frac{4}{9} \cdot \cos \left( \frac{2\pi}{p} \cdot 2^i \right) \right) \geq \exp \left( -\frac{4\pi^2}{p^2} \sum_{i=0}^{r-1} 2^{2i} \right) = \exp \left( -\frac{4\pi^2}{p^2} \cdot \frac{2^{2r} - 1}{3} \right) = \Omega(1)$$

which implies that  $a$  should be  $\Omega(r)$  for  $\hat{f}(1)$  to be polynomially small in  $p$ . Thus the extra  $\log p$  factor is necessary in this case and our analysis is tight. Note that in the special case of  $c = 1/2$ , the factors in the expressions for  $\hat{f}(s)$  are of the form  $\frac{1}{2} + \frac{1}{2} \cdot \omega^{2^i \cdot s}$  which can be arbitrarily close to zero. We remark that the construction, as stated, relies on there being infinitely many Mersenne primes. However, it seems to suggest that the extra  $\log p$  factor is necessary. Furthermore, similar examples can be produced with  $p$  that are not Mersenne, as long as the order of 2 modulo  $p$  is relatively small.

We now deduce several simple consequences of our lemma on random binary expansions that are used in the analysis of our reductions.

**Lemma 4.0.3.** *Let  $p > 2$  be prime. Suppose that  $c \leq q_1, q_2, \dots, q_t \leq 1 - c$  for some  $c \in (0, 1/2]$  and that  $Z_i \sim \text{Ber}(q_i)$  are independent. Let  $Y = \sum_{i=0}^t Z_i \cdot 2^i$  and for each  $x \in \mathbb{F}_p$ , let  $Y_x \sim \mathcal{L}(Y | Y \equiv x \pmod{p})$ . Consider  $Y_R$ , where  $R$  is chosen uniformly at random with  $R \sim \text{Unif}[\mathbb{F}_p]$ . If  $S = Y \pmod{p}$  is as in the previous lemma and  $\Delta = d_{TV}(\mathcal{L}(S), \text{Unif}[\mathbb{F}_p]) < p^{-1}$ , then it holds that*

1.  $d_{TV}(\mathcal{L}(Y), \mathcal{L}(Y_R)) \leq \Delta$ .
2. Given  $x \in \mathbb{F}_p$ , we may sample  $\mathcal{L}(Y_x)$  within  $\delta$  total variation distance in  $O\left(\frac{t \log(1/\delta)}{p^{-1} - \Delta}\right)$  time.

*Proof.* Note that the  $x \rightarrow Y_x$  defines a Markov transition sending  $S \rightarrow Y$  and  $R \rightarrow Y_R$ . The data-processing inequality yields  $d_{TV}(\mathcal{L}(Y), \mathcal{L}(Y_R)) \leq d_{TV}(\mathcal{L}(S), \mathcal{L}(R)) = \Delta$ , implying the first item.

The second item can be achieved by rejection sampling from the distribution  $\mathcal{L}(Y)$  until receiving an element congruent to  $x$  modulo  $p$  or reaching the cutoff of

$$m = \left\lceil \frac{\log \delta}{\log(1 - p^{-1} + \Delta)} \right\rceil = O\left(\frac{\log(1/\delta)}{p^{-1} - \Delta}\right)$$

rounds. Each sample from  $\mathcal{L}(Y)$  can be obtained in  $O(t)$  by sampling  $Z_0, Z_1, \dots, Z_t$  and forming the number  $Y$  with binary digits  $Z_t, Z_{t-1}, \dots, Z_0$ . If we receive a sample by the  $m$ th round, then it is exactly sampled from the conditional distribution  $\mathcal{L}(Y_x) = \mathcal{L}(Y|Y \equiv x \pmod{p})$ . Therefore the total variation between the output of this algorithm and  $\mathcal{L}(Y_x)$  is upper bounded by the probability that the rejection sampling scheme fails to output a sample. Now note that the probability that a sample is output in a single round is

$$\mathbb{P}[S = x] \geq p^{-1} - d_{\text{TV}}(\mathcal{L}(S), \text{Unif}[\mathbb{F}_p]) = p^{-1} - \Delta$$

by the definition of total variation. By the independence of sampling in different rounds, the probability that no sample is output is at most

$$(1 - \mathbb{P}[S = x])^m \leq (1 - p^{-1} + \Delta)^m \leq \delta$$

which completes the proof of the second item.  $\square$

We conclude this section with a sampling result similar to Lemma 4.0.3, but for the  $p = 2$  case.

**Lemma 4.0.4.** *Let  $R \sim \text{Unif}[\mathbb{F}_2]$ , and let  $\epsilon > 0$  and  $c \in (0, 1)$ . Then there exists  $t = O(c^{-1}(1-c)^{-1} \log(1/\epsilon))$ , so that in  $O(t \log(1/\epsilon))$  time one may sample  $X_1, \dots, X_t$  supported on  $\{0, 1\}$ , such that  $R = \sum_{i=1}^t X_i \pmod{2}$ , and such that  $d_{\text{TV}}(\mathcal{L}(X), \text{Ber}(c)^{\otimes t}) < \epsilon$ .*

*Proof.* Let  $Z_1, \dots, Z_t \stackrel{i.i.d}{\sim} \text{Ber}(c)$ . By induction on  $t$ , one may show that

$$\mathbb{P} \left[ \sum_{i=1}^t Z_i \equiv 0 \pmod{2} \right] = \frac{1}{2} - \frac{(1-2c)^t}{2}$$

Let  $t = \lceil \log(\epsilon/8) / \log(|1-2c|) \rceil + 1 = O(c^{-1}(1-c)^{-1} \log(1/\epsilon))$ , so that  $d_{\text{TV}}(\mathcal{L}(\sum_{i=1}^t Z_i), \mathcal{L}(R)) \leq$

$\epsilon/8$ . Sample the distribution

$$X \sim \mathcal{L} \left( Z \mid \sum_{i=1}^t Z_i \equiv R \pmod{2} \right)$$

within  $\epsilon/2$  total variation distance by rejection sampling. This takes time  $O(t \log(1/\epsilon))$ , because it consists of at most  $O(\log(1/\epsilon))$  rounds of sampling fresh copies of  $Z \sim \text{Ber}(c)^{\otimes t}$  and checking if  $\sum_{i=1}^t Z_i = R$ . By triangle inequality, it suffices to show that  $d_{\text{TV}}(\mathcal{L}(X), \text{Ber}(c)^{\otimes t}) \leq \epsilon/2$ . This is true because for any  $\omega \in \{0, 1\}^t$ ,

$$\begin{aligned} \mathbb{P}[X = \omega] &= \mathbb{P} \left[ Z = \omega \mid \sum_{i=1}^t Z_i \equiv \sum_{i=1}^t \omega_i \pmod{2} \right] \cdot \mathbb{P} \left[ R \equiv \sum_{i=1}^t \omega_i \right] \\ &= \frac{\mathbb{P}[Z = \omega]}{2 \cdot \mathbb{P} \left[ \sum_{i=1}^t Z_i \equiv \sum_{i=1}^t \omega_i \pmod{2} \right]}. \end{aligned}$$

Hence  $(1 - \epsilon/4)\mathbb{P}[Z = \omega] \leq \mathbb{P}[X = \omega] \leq (1 + \epsilon/4) \cdot \mathbb{P}[Z = \omega]$  for all  $\omega \in \{0, 1\}^t$ , and so  $d_{\text{TV}}(\mathcal{L}(X), \mathcal{L}(Z)) = d_{\text{TV}}(\mathcal{L}(X), \text{Ber}(c)^{\otimes t}) \leq \epsilon/2$ , as desired.  $\square$



# Chapter 5

## Algorithms for Counting $k$ -Cliques in $G(n, c, s)$

In this section, we consider several natural algorithms for counting  $k$ -cliques in  $G(n, c, s)$  with  $c = \Theta(n^{-\alpha})$  for some  $\alpha \in (0, 1)$ . The main objective of this section is to show that, when  $k$  and  $s$  are constant, these algorithms all run faster than all known algorithms for  $\#(k, s)$ -CLIQUE on worst-case hypergraphs and nearly match the lower bounds from our reduction for certain  $k, c$  and  $s$ . This demonstrates that the average-case complexity of  $\#(k, s)$ -CLIQUE on Erdős-Rényi hypergraphs is intrinsically different from its worst-case complexity. As discussed in Section 2.2, this also shows the necessity of a slowdown term comparable to  $\Upsilon_{\#}$  in our worst-case to average-case reduction for  $\#(k, s)$ -CLIQUE. We begin with a randomized sampling-based algorithm for counting  $k$ -cliques in  $G(n, c, s)$ , extending well-known greedy heuristics for finding  $k$ -cliques in random graphs. We then present an improvement to this algorithm in the graph case and a deterministic alternative.

### 5.1 Greedy Random Sampling

In this section, we consider a natural greedy algorithm GREEDY-RANDOM-SAMPLING for counting  $k$ -cliques in a  $s$ -uniform hypergraph  $G \sim G(n, c, s)$  with  $c = \Theta(n^{-\alpha})$ .

Given a subset of vertices  $A \subseteq [n]$  of  $G$ , define  $\text{CN}_G(A)$  to be

$$\text{CN}_G(A) = \{v \in V(G) \setminus A : B \cup \{v\} \in E(G) \text{ for all } (s-1)\text{-subsets } B \subseteq A\}$$

denote the set of common neighbors of the vertices in  $A$ . The algorithm GREEDY-RANDOM-SAMPLING maintains a set  $S$  of  $k$ -subsets of  $[n]$  and for  $T$  iterations does the following:

1. Sample distinct starting vertices  $v_1, v_2, \dots, v_{s-1}$  uniformly at random and proceed to sample the remaining vertices  $v_s, v_{s+1}, \dots, v_k$  iteratively so that  $v_{i+1}$  is chosen uniformly at random from  $\text{CN}_G(v_1, v_2, \dots, v_i)$  if it is nonempty.
2. If  $k$  vertices  $\{v_1, v_2, \dots, v_k\}$  are chosen then add  $\{v_1, v_2, \dots, v_k\}$  to  $S$  if it is not already in  $S$ .

This algorithm is an extension of the classical greedy algorithm for finding  $\log_2 n$  sized cliques in  $G(n, 1/2)$  in [54, 46], the Metropolis process examined in [51] and the greedy procedure solving  $k$ -CLIQUE on  $G(n, c)$  with  $c = \Theta(n^{-2/(k-1)})$  discussed by Rossman in [73]. These and other natural polynomial time search algorithms fail to find cliques of size  $(1 + \epsilon) \log_2 n$  in  $G(n, 1/2)$ , even though its clique number is approximately  $2 \log_2 n$  with high probability [61, 66]. Our algorithm GREEDY-RANDOM-SAMPLING extends this greedy algorithm to count  $k$ -cliques in  $G(n, c, s)$ . In our analysis, we will see a phase transition in the behavior of this algorithm at  $k = \tau$  for some  $\tau$  smaller than the clique number of  $G(n, c, s)$ . This is analogous to the breakdown of the natural greedy algorithm at cliques of size  $\log_2 n$  on  $G(n, 1/2)$ .

Before analyzing GREEDY-RANDOM-SAMPLING, we state a simple classical lemma counting the number of  $k$ -cliques in  $G(n, c, s)$ . This lemma follows from linearity of expectation and Markov's inequality.

**Lemma 5.1.1.** *For fixed  $\alpha \in (0, 1)$  and  $s$ , let  $\kappa \geq s$  be the largest positive integer satisfying  $\alpha \binom{\kappa}{s-1} < s$ . If  $G \sim G(n, c, s)$  where  $c = O(n^{-\alpha})$ , then  $\mathbb{E}[|\text{cl}_k(G)|] = \binom{n}{k} c^{\binom{k}{s}}$  and  $\omega(G) \leq \kappa + 1 + t$  with probability at least  $1 - O\left(n^{-\alpha t(1-s^{-1}) \binom{\kappa+2}{s-1}}\right)$  for any fixed nonnegative integer  $t$ .*



*Proof.* Let  $C > 0$  be such that  $c \leq Cn^{-\alpha}$  for sufficiently large  $n$ . For any given set  $\{v_1, v_2, \dots, v_k\}$  of  $k$  vertices in  $[n]$ , the probability that all hyperedges are present among  $\{v_1, v_2, \dots, v_k\}$  and thus these vertices form a  $k$ -clique in  $G$  is  $c^{\binom{k}{s}}$ . Linearity of expectation implies that the expected number of  $k$ -cliques is  $\mathbb{E}[|\text{cl}_k(G)|] = \binom{n}{k} c^{\binom{k}{s}}$ . Now consider taking  $k = \kappa + 2 + t$  and note that

$$\begin{aligned} \mathbb{E}[|\text{cl}_k(G)|] &= \binom{n}{k} c^{\binom{k}{s}} \leq n^k c^{\binom{k}{s}} \leq C^{\binom{k}{s}} \cdot \exp\left(\left(1 - \frac{\alpha}{s} \binom{k-1}{s-1}\right) k \log n\right) \\ &\leq C^{\binom{k}{s}} \cdot \exp\left(\left(1 - \frac{\alpha}{s} \binom{\kappa+1}{s-1}\right) k \log n - \frac{\alpha}{s} \cdot t \binom{\kappa+1}{s-2} k \log n\right) \\ &\leq C^{\binom{k}{s}} n^{-\alpha t (1-s^{-1}) \binom{\kappa+2}{s-1}} \end{aligned}$$

since  $k \geq \kappa + 2$  and  $\binom{\kappa+1+t}{s-1} \geq \binom{\kappa+1}{s-1} + t \binom{\kappa+1}{s-2}$  by iteratively applying Pascal's identity. Observe that  $\kappa = O(1)$  and thus  $C^{\binom{k}{s}} = O(1)$ . Now by Markov's inequality, it follows that  $\mathbb{P}[\omega(G) \geq k] = \mathbb{P}[|\text{cl}_k(G)| \geq 1] \leq \mathbb{E}[|\text{cl}_k(G)|]$ , completing the proof of the lemma.  $\square$

In particular, this implies that the clique number of  $G(n, c, s)$  is typically at most  $(s! \alpha^{-1})^{\frac{1}{s-1}} + s - 1$ . In the graph case of  $s = 2$ , this simplifies to  $1 + 2\alpha^{-1}$ . In the next subsection, we give upper bounds on the number of iterations  $T$  causing all  $k$ -cliques in  $G$  to end up in  $S$  and analyze the runtime of the algorithm. The subsequent subsection improves the runtime of GREEDY-RANDOM-SAMPLING for graphs when  $s = 2$  through a matrix multiplication post-processing step. The last subsection gives an alternative deterministic algorithm with a similar performance to GREEDY-RANDOM-SAMPLING.

## 5.2 Sample Complexity and Runtime of Greedy Random Sampling

In this section, we analyze the runtime of GREEDY-RANDOM-SAMPLING and prove upper bounds on the number of iterations  $T$  needed for the algorithm to terminate with  $S = \text{cl}_k(G)$ . The dynamic set  $S$  needs to support search and insertion of  $k$ -cliques. Consider labelling the vertices of  $G$  with elements of  $[n]$  and storing the

elements of  $S$  in a balanced binary search tree sorted according to the lexicographic order on  $[n]^k$ . Search and insertion can each be carried out in  $O(\log |\text{cl}_k(G)|) = O(k \log n)$  time. It follows that each iteration of GREEDY-RANDOM-SAMPLING therefore takes  $O(n + k \log n) = O(n)$  time as long as  $k = O(1)$ . Outputting  $|S|$  in GREEDY-RANDOM-SAMPLING therefore yields a  $O(nT)$  time algorithm for  $\#(k, s)$ -CLIQUE on  $G(n, c, s)$  that succeeds with high probability.

We now prove upper bounds on the minimum number of iterations  $T$  needed for this algorithm to terminate with  $S = \text{cl}_k(G)$  and therefore solve  $\#(k, s)$ -CLIQUE.

**Theorem 5.2.1.** *Let  $k$  and  $s$  be constants and  $c = \Theta(n^{-\alpha})$  for some  $\alpha \in (0, 1)$ . Let  $\tau$  be the largest integer satisfying  $\alpha \binom{\tau}{s-1} < 1$  and suppose that*

$$T \geq \begin{cases} 2n^{\tau+1} c^{\binom{\tau+1}{s}} (\log n)^{3(k-\tau)(1+\epsilon)} & \text{if } k \geq \tau + 1 \\ 2n^k c^{\binom{k}{s}} (\log n)^{1+\epsilon} & \text{if } k < \tau + 1 \end{cases}$$

for some  $\epsilon > 0$ . Then GREEDY-RANDOM-SAMPLING run with  $T$  iterations terminates with  $S = \text{cl}_k(G)$  with probability  $1 - n^{-\omega(1)}$  over the random bits of GREEDY-RANDOM-SAMPLING and with probability  $1 - n^{-\omega(1)}$  over the choice of random hypergraph  $G \sim G(n, c, s)$ .

*Proof.* We first consider the case where  $k \geq \tau + 1$ . Fix some  $\epsilon > 0$  and let  $v = (v_1, v_2, \dots, v_k)$  be an ordered tuple of distinct vertices in  $[n]$ . Define the random variable

$$Z_v = n(n-1) \cdots (n-s+2) \prod_{i=s-1}^{k-1} |\text{CN}_G(v_1, v_2, \dots, v_i)|$$

Consider the following event over the sampling  $G \sim G(n, c, s)$

$$A_v = \left\{ Z_v \geq 2n^{\tau+1} c^{\binom{\tau+1}{s}} (\log n)^{3(k-1-\tau)(1+\epsilon)} \quad \text{and} \quad \{v_1, v_2, \dots, v_k\} \in \text{cl}_k(G) \right\}$$

We now proceed to bound the probability of  $A_v$  through simple Chernoff and union bounds over  $G$ . In the next part of the argument, we condition on the event that  $\{v_1, v_2, \dots, v_k\}$  forms a clique in  $G$ . For each  $i \in \{s-1, s, \dots, k-1\}$ , let  $Y_{v,i}$  be the number of common neighbors of  $v_1, v_2, \dots, v_i$  in  $V(G) \setminus \{v_1, v_2, \dots, v_k\}$ . Note that  $Y_{v,i} \sim \text{Bin}\left(n-k, c^{\binom{i}{s-1}}\right)$  and that  $|\text{CN}_G(v_1, v_2, \dots, v_i)| = k-i + Y_{v,i}$ . The standard

Chernoff bound for the binomial distribution implies that for all  $\delta_i > 0$ ,

$$\mathbb{P} \left[ |\text{CN}_G(v_1, v_2, \dots, v_i)| \geq k - i + (1 + \delta_i)(n - k)c^{\binom{i}{s-1}} \right] \leq \exp \left( -\frac{\delta_i^2}{2 + \delta_i} \cdot (n - k)c^{\binom{i}{s-1}} \right)$$

Now define  $\kappa_i$  to be

$$\kappa_i = (n - k)^{-1} c^{-\binom{i}{s-1}} \cdot (\log n)^{1+\epsilon}$$

for each  $i \in \{s - 1, s, \dots, k - 1\}$ . Let  $\delta_i = \sqrt{\kappa_i}$  if  $i \leq \tau$  and  $\delta_i = \kappa_i$  if  $i > \tau$ . Note that for sufficiently large  $n$ ,  $\delta_i < 1$  if  $i \leq \tau$  and  $\delta_i \geq 1$  if  $i > \tau$ . These choices of  $\delta_i$  ensure that the Chernoff upper bounds above are each at most  $\exp(-\frac{1}{3}(\log n)^{1+\epsilon})$  for each  $i$ . A union bound implies that with probability at least  $1 - k \exp(-\frac{1}{3}(\log n)^{1+\epsilon})$ , it holds that

$$|\text{CN}_G(v_1, v_2, \dots, v_i)| < k - i + (1 + \delta_i)(n - k)c^{\binom{i}{s-1}} < (1 + 2\delta_i)(n - k)c^{\binom{i}{s-1}}$$

for all  $i$  and sufficiently large  $n$ . Here, we used the fact that  $\delta_i(n - k)c^{\binom{i}{s-1}} = \omega(1)$  for all  $i$  by construction and  $k = O(1)$ . Observe that  $(1 + 2\delta_i)(n - k)c^{\binom{i}{s-1}} \leq 3(\log n)^{1+\epsilon}$  for all  $i \geq \tau + 1$ . These inequalities imply that

$$\begin{aligned} \log Z_v &< \log n^{s-1} + \sum_{i=s-1}^{\tau} \log \left( (1 + 2\delta_i)(n - k)c^{\binom{i}{s-1}} \right) + 3(k - 1 - \tau)(1 + \epsilon) \log \log n \\ &< \log n^{\tau+1} + (\log c) \sum_{i=s-1}^{\tau} \binom{i}{s-1} + \sum_{i=s-1}^{\tau} \log(1 + 2\delta_i) + 3(k - 1 - \tau)(1 + \epsilon) \log \log n \\ &\leq \log \left( n^{\tau+1} c^{\binom{\tau+1}{s}} \right) + 3(k - 1 - \tau)(1 + \epsilon) \log \log n + 2 \sum_{i=s-1}^{\tau} \delta_i \\ &\leq \log \left( n^{\tau+1} c^{\binom{\tau+1}{s}} \right) + 3(k - 1 - \tau)(1 + \epsilon) \log \log n + o(1) \end{aligned}$$

The last inequality holds since  $\tau = O(1)$  and since  $\delta_i \lesssim (\log n)^{\frac{1}{2} + \frac{\epsilon}{2}} n^{-\frac{1}{2} + \frac{1}{2}\alpha \binom{\tau}{s-1}} = o(1)$  for all  $i \leq \tau$  because of the definition that  $\alpha \binom{\tau}{s-1} < 1$ . In summary, we have shown that for sufficiently large  $n$

$$\mathbb{P} \left[ Z_v \geq 2n^{\tau+1} c^{\binom{\tau+1}{s}} (\log n)^{3(k-1-\tau)(1+\epsilon)} \mid \{v_1, v_2, \dots, v_k\} \in \text{cl}_k(G) \right] \leq k \exp \left( -\frac{1}{3}(\log n)^{1+\epsilon} \right) = n^{-\omega(1)}$$

for any  $k$ -tuple of vertices  $v = (v_1, v_2, \dots, v_k)$ . Now since  $\mathbb{P}[\{v_1, v_2, \dots, v_k\} \in \text{cl}_k(G)] = c^{\binom{k}{s}}$ , we have that  $\mathbb{P}[A_v] \leq c^{\binom{k}{s}} n^{-\omega(1)} = n^{-\omega(1)}$  for each  $k$ -tuple  $v$ . Now consider the event

$$B = \left\{ Z_v < 2n^{\tau+1} c^{\binom{\tau+1}{s}} (\log n)^{3(k-1-\tau)(1+\epsilon)} \text{ for all } v \text{ such that } \{v_1, v_2, \dots, v_k\} \in \text{cl}_k(G) \right\}$$

Note that  $\overline{B} = \bigcup_{k\text{-tuples } v} A_v$  and thus a union bound implies that  $\mathbb{P}[B] \geq 1 - \sum_v \mathbb{P}[A_v] \geq 1 - n^k \cdot n^{-\omega(1)} = 1 - n^{-\omega(1)}$  since there are fewer than  $n^k$   $k$ -tuples  $v$ .

We now show that as long as  $B$  holds over the random choice of  $G$ , then the algorithm GREEDY-RANDOM-SAMPLING terminates with  $S = \text{cl}_k(G)$  with probability  $1 - n^{-\omega(1)}$  over the random bits of GREEDY-RANDOM-SAMPLING, which completes the proof of the lemma in the case  $k > \tau + 1$ . In the next part of the argument, we consider  $G$  conditioned on the event  $B$ . Fix some ordering  $v = (v_1, v_2, \dots, v_k)$  of some  $k$ -clique  $\mathcal{C} = \{v_1, v_2, \dots, v_k\}$  in  $G$ . Note that in any one of the  $T$  iterations of GREEDY-RANDOM-SAMPLING, the probability that the  $k$  vertices  $v_1, v_2, \dots, v_k$  are chosen in that order is exactly  $1/Z_v$ . Since the  $T$  iterations of GREEDY-RANDOM-SAMPLING are independent, we have that

$$\mathbb{P}[v \text{ is never chosen in a round}] = \left(1 - \frac{1}{Z_v}\right)^T \leq \exp\left(-\frac{T}{Z_v}\right) = n^{-\omega(1)}$$

since  $T$  is chosen so that  $T \geq Z_v (\log n)^{3(1+\epsilon)}$  for all  $k$ -tuples  $v$ , given the event  $B$ . Since there are at most  $n^k$  possible  $v$ , a union bound implies that every such  $v$  is chosen in a round of GREEDY-RANDOM-SAMPLING with probability at least  $1 - n^k \cdot n^{-\omega(1)} = 1 - n^{-\omega(1)}$  over the random bits of the algorithm. In this case,  $S = \text{cl}_k(G)$  after the  $T$  rounds of GREEDY-RANDOM-SAMPLING. This completes the proof of the theorem in the case  $k \geq \tau + 1$ .

We now handle the case  $k < \tau + 1$  through a nearly identical argument. Define  $\kappa_i$  as in the previous case and set  $\delta_i = \sqrt{\kappa_i}$  for all  $i \in \{s-1, s, \dots, k-1\}$ . By the same argument, for each  $k$ -tuple  $v$  we have with probability  $1 - n^{-\omega(1)}$  over the choice of

$G$  that

$$\begin{aligned}
\log Z_v &< \log n^{s-1} + \sum_{i=s-1}^{k-1} \log \left( (1 + 2\delta_i)(n - k)c^{\binom{i}{s-1}} \right) \\
&< \log n^k + (\log c) \sum_{i=s-1}^{k-1} \binom{i}{s-1} + 2 \sum_{i=s-1}^{k-1} \delta_i \\
&= \log \left( n^k c^{\binom{k}{s}} \right) + o(1)
\end{aligned}$$

where again  $\delta_i \lesssim (\log n)^{\frac{1}{2} + \frac{\epsilon}{2}} n^{-\frac{1}{2} + \frac{1}{2}\alpha \binom{\tau}{s-1}} = o(1)$  for all  $i \leq k-1 < \tau$ . Define the event

$$B' = \left\{ Z_v < 2n^k c^{\binom{k}{s}} (\log n)^{1+\epsilon} \text{ for all } v \text{ such that } \{v_1, v_2, \dots, v_k\} \in \text{cl}_k(G) \right\}$$

Note that  $T$  is such that  $T \geq Z_v (\log n)^{1+\epsilon}$  for all  $v$  if  $B'$  holds. Now repeating the rest of the argument from the  $k \geq \tau + 1$  case shows that  $\mathbb{P}[B'] \geq 1 - n^{-\omega(1)}$  and that GREEDY-RANDOM-SAMPLING terminates with  $S = \text{cl}_k(G)$  with probability  $1 - n^{-\omega(1)}$  over its random bits if  $G$  is such that  $B'$  holds. This completes the proof of the theorem.  $\square$

Implementing  $S$  as a balanced binary search tree and outputting  $|S|$  in GREEDY-RANDOM-SAMPLING therefore yields the following algorithmic upper bounds for  $\#(k, s)$ -CLIQUE with inputs sampled from  $G(n, c, s)$ .

**Corollary 5.2.2.** *Suppose that  $k$  and  $s$  are constants and  $c = \Theta(n^{-\alpha})$  for some  $\alpha \in (0, 1)$ . Let  $\tau$  be the largest integer satisfying  $\alpha \binom{\tau}{s-1} < 1$ . Then it follows that*

1. *If  $k \geq \tau + 1$ , there is an  $\tilde{O} \left( n^{\tau+2-\alpha \binom{\tau+1}{s}} \right)$  time randomized algorithm solving  $\#(k, s)$ -CLIQUE on inputs sampled from  $G(n, c, s)$  with probability at least  $1 - n^{-\omega(1)}$ .*
2. *If  $k < \tau + 1$ , there is an  $\tilde{O} \left( n^{k+1-\alpha \binom{k}{s}} \right)$  time randomized algorithm solving  $\#(k, s)$ -CLIQUE on inputs sampled from  $G(n, c, s)$  with probability at least  $1 - n^{-\omega(1)}$ .*

By Lemma 5.1.1, the hypergraph  $G \sim G(n, c, s)$  has clique number  $\omega(G) \leq \kappa + 2$  with probability  $1 - 1/\text{poly}(n)$  if where  $\kappa \geq s$  is the largest positive integer satisfying

$\alpha \binom{\kappa}{s-1} < s$ . In particular, when  $k > \kappa + 2$  in the theorem above, the algorithm outputting zero succeeds with probability  $1 - 1/\text{poly}(n)$  and  $\#(k, s)$ -CLIQUE is trivial. For there to typically be a nonzero number of  $k$ -cliques in  $G(n, c, s)$ , it should hold that  $0 < \alpha \leq s \binom{k-1}{s-1}^{-1}$ . In the graph case of  $s = 2$ , this simplifies to the familiar condition that  $0 < \alpha \leq \frac{2}{k-1}$ . We also remark that when  $k < \tau + 1$ , the runtime of this algorithm is an  $\tilde{O}(n)$  factor off from the expected number of  $k$ -cliques in  $G \sim G(n, c, s)$ .

### 5.3 Post-Processing with Matrix Multiplication

In this section, we improve the runtime of GREEDY-RANDOM-SAMPLING as an algorithm for  $\#(k, s)$ -CLIQUE in the graph case of  $s = 2$ . The improvement comes from the matrix multiplication step of Nešetřil and Poljak from their  $O(n^{\omega \lceil k/3 \rceil + (k \pmod{3})})$  time worst-case algorithm for  $\#(k, 2)$ -CLIQUE [64]. Our improved runtime for GREEDY-RANDOM-SAMPLING is stated in the following theorem.

**Theorem 5.3.1.** *Suppose that  $k > 2$  is a fixed positive integer and  $c = \Theta(n^{-\alpha})$  where  $0 < \alpha \leq \frac{2}{k-1}$  is also fixed. Then there is a randomized algorithm solving  $\#(k, 2)$ -CLIQUE on inputs sampled from  $G(n, c)$  with probability  $1 - n^{-\omega(1)}$  that runs in  $\tilde{O}\left(n^{\omega \lceil k/3 \rceil + \omega - \omega \alpha \binom{\lceil k/3 \rceil}{2}}\right)$  time.*

*Proof.* Label the vertices of an input graph  $G \sim G(n, c)$  with the elements of  $[n]$ . Consider the following application of GREEDY-RANDOM-SAMPLING with post-processing:

1. Run GREEDY-RANDOM-SAMPLING to compute the two sets of cliques  $S_1 = \text{cl}_{\lceil k/3 \rceil}(G)$  and  $S_2 = \text{cl}_{\lfloor k/3 \rfloor}(G)$  with the number of iterations  $T$  as given in Theorem 5.2.1.
2. Construct the matrix  $M_1 \in \{0, 1\}^{|S_1| \times |S_1|}$  with rows and columns indexed by the elements of  $S_1$  such that  $(M_1)_{A,B} = 1$  for  $A, B \in S_1$  if  $A \cup B$  forms a clique of  $G$  and all labels in  $A$  are strictly less than all labels in  $B$ .
3. Construct the matrix  $M_2 \in \{0, 1\}^{|S_1| \times |S_2|}$  with rows indexed by the elements of  $S_1$  and columns indexed by the elements of  $S_2$  such that  $(M_2)_{A,B} = 1$  for  $A \in S_1$

and  $B \in S_2$  under the same rule that  $A \cup B$  forms a clique of  $G$  and all labels in  $A$  are strictly less than all labels in  $B$ . Construct the matrix  $M_3$  with rows and columns indexed by  $S_2$  analogously.

4. Compute the matrix product

$$M_P = \begin{cases} M_1^2 & \text{if } k \equiv 0 \pmod{3} \\ M_1 M_2 & \text{if } k \equiv 1 \pmod{3} \\ M_2 M_3 & \text{if } k \equiv 2 \pmod{3} \end{cases}$$

5. Output the sum of entries

$$\sum_{(A,B) \in \mathcal{S}} (M_P)_{A,B}$$

where  $\mathcal{S}$  is the support of  $M_1$  if  $k \equiv 0 \pmod{3}$  and  $\mathcal{S}$  is the support of  $M_2$  if  $k \not\equiv 0 \pmod{3}$ .

We will show that this algorithm solves  $\#(k, 2)$ -CLIQUE with probability  $1 - n^{-\omega(1)}$  when  $k \equiv 1 \pmod{3}$ . The cases when  $k \equiv 0, 2 \pmod{3}$  follow from a nearly identical argument. By Theorem 5.2.1, the first step applying GREEDY-RANDOM-SAMPLING succeeds with probability  $1 - n^{-\omega(1)}$ . Note that  $(M_P)_{A,B}$  counts the number of  $\lfloor k/3 \rfloor$ -cliques  $C$  in  $G$  such that the labels of  $C$  are strictly greater than those of  $A$  and less than those of  $B$  and such that  $A \cup C$  and  $C \cup B$  are both cliques. If it further holds that  $(M_2)_{A,B} = 1$ , then  $A \cup B$  is a clique and  $A \cup B \cup C$  is also clique. Therefore the sum output by the algorithm exactly counts the number of triples  $(A, B, C)$  such that  $A \cup B \cup C$  is a clique,  $|A| = |C| = \lfloor k/3 \rfloor$ ,  $|B| = \lceil k/3 \rceil$  and the labels of  $C$  are greater than those of  $A$  and less than those of  $B$ . Observe that any clique  $\mathcal{C} \in \text{cl}_k(G)$  is counted in this sum exactly once by the triple  $(A, B, C)$  where  $A$  consists of the lowest  $\lfloor k/3 \rfloor$  labels in  $\mathcal{C}$ ,  $B$  consists of the highest  $\lceil k/3 \rceil$  labels in  $\mathcal{C}$  and  $C$  contains the remaining vertices of  $\mathcal{C}$ . Therefore this algorithm solves  $\#(k, 2)$ -CLIQUE as long as Step 1 succeeds.

It suffices to analyze the additional runtime incurred by this post-processing. Observe that the number of cliques output by a call to greedy-random-sampling with

$T$  iterations is at most  $T$ . Also note that if  $\alpha \leq \frac{2}{k-1}$ , then  $\tau \geq \lfloor \frac{k}{2} \rfloor - 1$ . If  $k \geq 3$ , then it follows that  $\tau + 1 \geq \lfloor \frac{k}{2} \rfloor \geq \lceil \frac{k}{3} \rceil$ . It follows by Theorem 5.2.1 that  $\max\{|S_1|, |S_2|\} = \tilde{O}\left(n^{\lceil k/3 \rceil + 1 - \alpha \binom{\lceil k/3 \rceil}{s}}\right)$ . Note that computing the matrix  $M_P$  takes  $\tilde{O}(\max\{|S_1|, |S_2|\}^\omega) = \tilde{O}\left(n^{\omega \lceil k/3 \rceil + \omega - \omega \alpha \binom{\lceil k/3 \rceil}{s}}\right)$  time. Now observe that all other steps of the algorithm run in  $\tilde{O}\left(n^{2 \lceil k/3 \rceil - 2\alpha \binom{\lceil k/3 \rceil}{s}}\right)$  time, which completes the proof of the theorem since the matrix multiplication constant satisfies  $\omega \geq 2$ .  $\square$

We remark that for simplicity, we have ignored minor improvements in the runtime that can be achieved by more carefully analyzing Step 4 in terms of rectangular matrix multiplication constants if  $k \not\equiv 0 \pmod{3}$ . Note that the proof above implicitly used a weak large deviations bound on  $|\text{cl}_k(G)|$ . More precisely, it used the fact that if GREEDY-RANDOM-SAMPLING with  $T$  iterations succeeds, then  $|\text{cl}_k(G)| \leq T$ . Theorem 5.2.1 thus implies that  $|\text{cl}_k(G)|$  is upper bounded by the minimal settings of  $T$  in the theorem statement with probability  $1 - n^{-\omega(1)}$  over  $G \sim G(n, c, s)$ .

When  $k \leq \tau + 1$ , these upper bounds are a polylog( $n$ ) factor from the expectation of  $|\text{cl}_k(G)|$ . The upper tails of  $|\text{cl}_k(G)|$  and more generally of the counts of small subhypergraphs in  $G(n, c, s)$  have been studied extensively in the literature. We refer to [76, 50, 49, 26] for a survey of the area and recent results. Given a hypergraph  $H$ , let  $N(n, m, H)$  denote the largest number of copies of  $H$  that can be constructed in an  $s$ -uniform hypergraph with at most  $n$  vertices and  $m$  hyperedges. Define the quantity

$$M_H(n, c) = \max \left\{ m \leq \binom{n}{s} : \forall H' \subseteq H \text{ it holds that } N(n, m, H') \leq n^{|V(H')|} c^{|E(H')|} \right\}$$

The following large deviations result from [31] generalizes a graph large deviations bound from [49] to hypergraphs to obtain the following result.

**Theorem 5.3.2** (Theorem 4.1 from [31]). *For every  $s$ -uniform hypergraph  $H$  and every fixed  $\epsilon > 0$ , there is a constant  $C(\epsilon, H)$  such that for all  $n \geq |V(H)|$  and  $c \in (0, 1)$ , it holds that*

$$\mathbb{P}[X_H \geq (1 + \epsilon)\mathbb{E}[X_H]] \leq \exp(-C(\epsilon, H) \cdot M_H(n, c))$$



where  $X_H$  is the number of copies of  $H$  in  $G \sim G(n, c, s)$ .

Proposition 4.3 in [31] shows that if  $H$  is a  $d$ -regular  $s$ -uniform hypergraph and  $c \geq n^{-s/d}$  then  $M_H(n, c) = \Theta(n^s c^d)$ . This implies that

$$\mathbb{P} \left[ |\text{cl}_k(G)| \geq (1 + \epsilon) \binom{n}{k} c^{\binom{k}{s}} \right] \leq \exp \left( -C'(\epsilon) \cdot n^s c^{\binom{k-1}{s-1}} \right)$$

as long as  $c \geq n^{-s!(k-s)!/(k-1)!}$ . This provides strong bounds on the upper tails of  $|\text{cl}_k(G)|$  that will be useful in the next subsection.

## 5.4 Deterministic Iterative Algorithm for Counting in $G(n, c, s)$

In this section, we present an alternative deterministic algorithm IT-GEN-CLIQUEs achieving a similar runtime to GREEDY-RANDOM-SAMPLING. Although they have very different analyses, the algorithm IT-GEN-CLIQUEs can be viewed as a deterministic analogue of GREEDY-RANDOM-SAMPLING. Both are constructing cliques one vertex at a time. The algorithm IT-GEN-CLIQUEs takes in cutoffs  $C_{s-1}, C_s, \dots, C_k$  and generates sets  $S_{s-1}, S_s, \dots, S_k$  as follows:

1. Initialize  $S_{s-1}$  to be the set of all  $(s-1)$ -subsets of  $[n]$ .
2. Given the set  $S_i$ , for each vertex  $v \in [n]$ , iterate through all subsets  $A \in S_i$  and add  $A \cup \{v\}$  to  $S_{i+1}$  if  $A \cup \{v\}$  is a clique and  $v$  is larger than the labels of all of the vertices in  $A$ . Stop if ever  $|S_{i+1}| \geq C_{i+1}$ .
3. Stop once  $S_k$  has been generated and output  $S_k$ .

Suppose that  $C_t$  are chosen to be any high probability upper bounds on the number of  $t$ -cliques in  $G \sim G(n, c, s)$  such as the bounds in Theorem 5.3.2. Then we have the following guarantees for the algorithm IT-GEN-CLIQUEs.

**Theorem 5.4.1.** *Suppose that  $k$  and  $s$  are constants and  $c = \Theta(n^{-\alpha})$  for some  $\alpha \in (0, 1)$ . Let  $\tau$  be the largest integer satisfying  $\alpha \binom{\tau}{s-1} < 1$  and  $C_t = 2n^t c^{\binom{t}{s}}$  for*

each  $s \leq t \leq k$ . Then IT-GEN-CLIQUEs with the cutoffs  $C_t$  outputs  $S_k = \text{cl}_k(G)$  with probability  $1 - n^{-\omega(1)}$  where

1. The runtime of IT-GEN-CLIQUEs is  $O\left(n^{\tau+2-\alpha\binom{\tau+1}{s}}\right)$  if  $k \geq \tau + 2$ .
2. The runtime of IT-GEN-CLIQUEs is  $O\left(n^{k-\alpha\binom{k-1}{s}}\right)$  if  $k < \tau + 2$ .

*Proof.* We first show that  $S_k = \text{cl}_k(G)$  with probability  $1 - n^{-\omega(1)}$  in IT-GEN-CLIQUEs. By a union bound and Theorem 5.3.2, it follows that  $|\text{cl}_t(G)| < C_t$  for each  $s \leq t \leq k$  with probability at least  $1 - (k-s+1)n^{-\omega(1)}$ . The following simple induction argument shows that  $S_t = \text{cl}_t(G)$  for each  $s-1 \leq t \leq k$  conditioned on this event. Note that  $\text{cl}_{s-1}(G)$  is by definition the set of all  $(s-1)$ -subsets of  $[n]$  and thus  $S_{s-1} = \text{cl}_{s-1}(G)$ . If  $S_t = \text{cl}_t(G)$ , then each  $(t+1)$ -clique  $\mathcal{C}$  of  $G$  is added exactly once to  $S_{t+1}$  as  $A \cup \{v\}$  where  $v$  is the vertex of  $\mathcal{C}$  with the largest label and  $A = \mathcal{C} \setminus \{v\} \in \text{cl}_t(G)$  are the remaining vertices. Now note that the runtime of IT-GEN-CLIQUEs is

$$O\left(\sum_{t=s-1}^{k-1} nC_t\right) = O\left(\max_{s-1 \leq t \leq k-1} (nC_t)\right) = \begin{cases} O\left(n^{\tau+2-\alpha\binom{\tau+1}{s}}\right) & \text{if } k \geq \tau + 2 \\ O\left(n^{k-\alpha\binom{k-1}{s}}\right) & \text{if } k < \tau + 2 \end{cases}$$

since  $k = O(1)$ . To see the second inequality, note that  $\log_n(C_{t+1}/C_t) = 1 - \alpha\binom{t}{s-1}$ . This implies that  $C_{t+1} > C_t$  if  $t \leq \tau$  and  $C_t$  is maximized on  $s \leq t \leq k$  when  $t = \tau + 1$ . This completes the proof of the theorem.  $\square$

We remark that in the case of  $k < \tau + 1$ , IT-GEN-CLIQUEs attains a small runtime improvement over GREEDY-RANDOM-SAMPLING. However, GREEDY-RANDOM-SAMPLING can be modified to match this runtime up to a polylog( $n$ ) factor by instead generating the  $(k-1)$ -cliques of  $G$  and applying the last step of IT-GEN-CLIQUEs to generate the  $k$ -cliques of  $G$ . We also remark that IT-GEN-CLIQUEs can also be used instead of GREEDY-RANDOM-SAMPLING in Step 1 of the algorithm in Theorem 5.3.1, yielding a nearly identical runtime of  $\tilde{O}\left(n^{\omega\lceil k/3 \rceil - \omega\alpha\binom{\lceil k/3 \rceil - 1}{2}}\right)$  for  $\#(k, 2)$ -CLIQUE on inputs sampled from  $G(n, c)$ .

# Chapter 6

## Extensions and Open Problems

In this chapter, we outline several extensions of our methods and problems left open after our work.

**Improved Average-Case Lower Bounds.** A natural question is whether tight average-case lower bounds for  $\#(k, s)$ -CLIQUE can be shown above the  $k$ -clique percolation threshold when  $s \geq 3$  and if the constant  $C$  in the exponent of our lower bounds for the graph case of  $s = 2$  can be improved from 1 to  $\omega/9$ .

**Raising Error Tolerance for Average-Case Hardness.** A natural question is whether the error tolerance of the worst-case to average-case reductions in Theorems 2.2.1 and 2.2.2 can be increased. We remarked in the introduction that for certain choices of  $k$ , the error tolerance cannot be significantly increased – for example, when  $k = 3 \log_2 n$ , the trivial algorithm that outputs 0 on any graph has subpolynomial error on graphs drawn from  $G(n, 1/2)$ , but is useless for reductions from worst-case graphs. Nevertheless, for other regimes of  $k$ , such as when  $k = O(1)$  is constant, counting  $k$ -cliques with error probability less than  $1/4$  on graphs drawn from  $G(n, 1/2)$  appears to be nontrivial. It is an open problem to prove hardness for such a regime. In general, one could hope to understand the tight tradeoffs between computation time, error tolerance,  $k$ ,  $c$ , and  $s$  for  $k$ -clique-counting on  $G(n, c, s)$ .

**Hardness of Approximating Clique Counts.** Another interesting question is whether it is hard to approximate the  $k$ -clique counts, within some additive error  $\epsilon$ , of hypergraphs drawn from  $G(n, c, s)$ . Since the number of  $k$ -cliques in  $G(n, c, s)$  concentrates around the mean  $\mu \approx c^{\binom{k}{s}} n^k$  with standard deviation  $\sigma$ , one would have to choose  $\epsilon \ll \sigma$  for approximation to be hard.

**Inhomogeneous Erdős-Rényi Hypergraphs.** Consider an inhomogeneous Erdős-Rényi hypergraph model, where each hyperedge  $e$  is independently chosen to be in the hypergraph with probability  $c(e)$ . Also suppose that we may bound  $c(e)$  uniformly away from 0 and 1 (that is,  $c(e) \in [c, 1 - c]$  for all possible hyperedges  $e$  and for some constant  $c$ ). We would like to prove that  $\#(k, s)$ -CLIQUE and PARITY- $(k, s)$ -CLIQUE are hard on average for inhomogeneous Erdős-Rényi hypergraphs. Unfortunately, this does not follow directly from our proof techniques because step 5 in the proof of Theorems 2.2.1 and 2.2.2 breaks down due to the inhomogeneity of the model. Nevertheless, steps 1-4 still hold, and therefore we can show that  $\#(k, s)$ -CLIQUE and PARITY- $(k, s)$ -CLIQUE are average-case hard for  $k$ -partite inhomogeneous Erdős-Rényi hypergraphs – when only the edges  $e$  that respect the  $k$ -partition are chosen to be in the hypergraph with inhomogeneous edge-dependent probability  $c(e) \in [c, 1 - c]$ .

# Bibliography

- [1] Miklós Ajtai. Generating hard instances of lattice problems. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 99–108. ACM, 1996.
- [2] Noga Alon and Ravi B Boppana. The monotone circuit complexity of boolean functions. *Combinatorica*, 7(1):1–22, 1987.
- [3] Noga Alon, Michael Krivelevich, and Benny Sudakov. Finding a large hidden clique in a random graph. *Random Structures and Algorithms*, 13(3-4):457–466, 1998.
- [4] Kazuyuki Amano and Akira Maruoka. A superpolynomial lower bound for a circuit computing the clique function with at most  $(1/6) \log \log n$  negation gates. *SIAM Journal on Computing*, 35(1):201–216, 2005.
- [5] Brendan PW Ames and Stephen A Vavasis. Nuclear norm minimization for the planted clique and biclique problems. *Mathematical programming*, 129(1):69–89, 2011.
- [6] Albert Atserias, Ilario Bonacina, Susanna F de Rezende, Massimo Lauria, Jakob Nordström, and Alexander Razborov. Clique is hard on average for regular resolution. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 866–877. ACM, 2018.
- [7] Eric Bach, James Driscoll, and Jeffrey Shallit. Factor refinement. *Journal of Algorithms*, 15(2):199–222, 1993.
- [8] Marshall Ball, Alon Rosen, Manuel Sabin, and Prashant Nalini Vasudevan. Average-case fine-grained hardness. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 483–496. ACM, 2017.
- [9] Marshall Ball, Alon Rosen, Manuel Sabin, and Prashant Nalini Vasudevan. Proofs of work from worst-case assumptions. In *Annual International Cryptology Conference*, pages 789–819. Springer, 2018.
- [10] Boaz Barak, Samuel B Hopkins, Jonathan Kelner, Pravesh Kothari, Ankur Moitra, and Aaron Potechin. A nearly tight sum-of-squares lower bound for the planted clique problem. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 428–437. IEEE, 2016.

- [11] Quentin Berthet and Philippe Rigollet. Complexity theoretic lower bounds for sparse principal component detection. In *COLT*, pages 1046–1066, 2013.
- [12] Andrej Bogdanov and Luca Trevisan. Average-case complexity. *Foundations and Trends® in Theoretical Computer Science*, 2(1):1–106, 2006.
- [13] Andrej Bogdanov and Luca Trevisan. On worst-case to average-case reductions for np problems. *SIAM Journal on Computing*, 36(4):1119–1159, 2006.
- [14] Enric Boix-Adserà, Matthew Brennan, and Guy Bresler. The average-case complexity of counting cliques in erdős-rényi hypergraphs. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1256–1280. IEEE, 2019.
- [15] Béla Bollobás and Oliver Riordan. Clique percolation. *Random Structures & Algorithms*, 35(3):294–322, 2009.
- [16] Matthew Brennan and Guy Bresler. Optimal average-case reductions to sparse pca: From weak assumptions to strong hardness. *arXiv preprint arXiv:1902.07380*, 2019.
- [17] Matthew Brennan, Guy Bresler, and Wasim Huleihel. Reducibility and computational lower bounds for problems with planted sparse structure. In *Conference On Learning Theory*, pages 48–166, 2018.
- [18] Matthew Brennan, Guy Bresler, and Wasim Huleihel. Universality of computational lower bounds for submatrix detection. *arXiv preprint arXiv:1902.06916*, 2019.
- [19] Jin-Yi Cai, Aduri Pavan, and D Sivakumar. On the hardness of permanent. In *Annual Symposium on Theoretical Aspects of Computer Science*, pages 90–99. Springer, 1999.
- [20] Jianer Chen, Xiuzhen Huang, Iyad A Kanj, and Ge Xia. Strong computational lower bounds via parameterized complexity. *Journal of Computer and System Sciences*, 72(8):1346–1367, 2006.
- [21] Yudong Chen. Incoherence-optimal matrix completion. *IEEE Transactions on Information Theory*, 61(5):2909–2923, 2015.
- [22] Yudong Chen and Jiaming Xu. Statistical-computational tradeoffs in planted problems and submatrix localization with a growing number of clusters and submatrices. *Journal of Machine Learning Research*, 17(27):1–57, 2016.
- [23] Amin Coja-Oghlan and Charilaos Efthymiou. On independent sets in random graphs. *Random Structures & Algorithms*, 47(3):436–486, 2015.
- [24] Mina Dalirrooyfard, Andrea Lincoln, and Virginia Vassilevska Williams. New techniques for proving fine-grained average-case hardness. *arXiv preprint arXiv:2008.06591*, 2020.

- [25] Yael Dekel, Ori Gurel-Gurevich, and Yuval Peres. Finding hidden cliques in linear time with high probability. *Combinatorics, Probability and Computing*, 23(1):29–49, 2014.
- [26] Robert DeMarco and Jeff Kahn. Tight upper tail bounds for cliques. *Random Structures & Algorithms*, 41(4):469–487, 2012.
- [27] Imre Derényi, Gergely Palla, and Tamás Vicsek. Clique percolation in random networks. *Physical review letters*, 94(16):160202, 2005.
- [28] Yash Deshpande and Andrea Montanari. Finding hidden cliques of size  $\sqrt{N/e}$  in nearly linear time. *Foundations of Computational Mathematics*, 15(4):1069–1128, 2015.
- [29] Sergey N Dorogovtsev, Alexander V Goltsev, and José FF Mendes. Critical phenomena in complex networks. *Reviews of Modern Physics*, 80(4):1275, 2008.
- [30] Rod G Downey and Michael R Fellows. Fixed-parameter tractability and completeness ii: On completeness for w [1]. *Theoretical Computer Science*, 141(1-2):109–131, 1995.
- [31] Andrzej Dudek, Joanna Polcyn, and Andrzej Ruciński. Subhypergraph counts in extremal and random hypergraphs and the fractional q-independence. *Journal of combinatorial optimization*, 19(2):184–199, 2010.
- [32] Uriel Feige, David Gamarnik, Joe Neeman, Miklós Z Rácz, and Prasad Tetali. Finding cliques using few probes. *arXiv preprint arXiv:1809.06950*, 2018.
- [33] Uriel Feige and Robert Krauthgamer. Finding and certifying a large hidden clique in a semirandom graph. *Random Structures and Algorithms*, 16(2):195–208, 2000.
- [34] Uriel Feige and Carsten Lund. On the hardness of computing the permanent of random matrices. In *Proceedings of the twenty-fourth annual ACM symposium on Theory of computing*, pages 643–654. ACM, 1992.
- [35] Uriel Feige and Dorit Ron. Finding hidden cliques in linear time. In *21st International Meeting on Probabilistic, Combinatorial, and Asymptotic Methods in the Analysis of Algorithms (AofA '10)*, pages 189–204. Discrete Mathematics and Theoretical Computer Science, 2010.
- [36] Joan Feigenbaum and Lance Fortnow. Random-self-reducibility of complete sets. *SIAM Journal on Computing*, 22(5):994–1005, 1993.
- [37] Vitaly Feldman, Elena Grigorescu, Lev Reyzin, Santosh Vempala, and Ying Xiao. Statistical algorithms and a lower bound for detecting planted cliques. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 655–664. ACM, 2013.

- [38] David Gamarnik. Computing the partition function of the sherrington-kirkpatrick model is hard on average. *arXiv preprint arXiv:1810.05907*, 2018.
- [39] David Gamarnik and Madhu Sudan. Limits of local algorithms over sparse random graphs. In *Proceedings of the 5th conference on Innovations in theoretical computer science*, pages 369–376. ACM, 2014.
- [40] Shuhong Gao. Normal bases over finite fields. *Doctoral thesis, Waterloo*, 1993.
- [41] Peter Gemmell, Richard Lipton, Ronitt Rubinfeld, Madhu Sudan, and Avi Wigderson. Self-testing/correcting for polynomials and for approximate functions. In *Proceedings of the twenty-third annual ACM symposium on Theory of computing*, pages 33–42. ACM, 1991.
- [42] Peter Gemmell and Madhu Sudan. Highly resilient correctors for polynomials. *Inf. Process. Lett.*, 43(4):169–174, 1992.
- [43] Oded Goldreich. On counting t-cliques mod 2. 2020.
- [44] Oded Goldreich and Guy Rothblum. Counting t-cliques: Worst-case to average-case reductions and direct interactive proof systems. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 77–88. IEEE, 2018.
- [45] Oded Goldreich and Guy N Rothblum. Worst-case to average-case reductions for subclasses of p. 2017.
- [46] Geoffrey R Grimmett and Colin JH McDiarmid. On colouring random graphs. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 77, pages 313–324. Cambridge University Press, 1975.
- [47] Bruce E Hajek, Yihong Wu, and Jiaming Xu. Computational lower bounds for community detection on random graphs. In *COLT*, pages 899–928, 2015.
- [48] Alon Itai and Michael Rodeh. Finding a minimum circuit in a graph. *SIAM Journal on Computing*, 7(4):413–423, 1978.
- [49] Svante Janson, Krzysztof Oleszkiewicz, and Andrzej Ruciński. Upper tails for subgraph counts in random graphs. *Israel Journal of Mathematics*, 142(1):61–92, 2004.
- [50] Svante Janson and Andrzej Ruciński. The infamous upper tail. *Random Structures & Algorithms*, 20(3):317–342, 2002.
- [51] Mark Jerrum. Large cliques elude the metropolis process. *Random Structures & Algorithms*, 3(4):347–359, 1992.
- [52] Ari Juels and Marcus Peinado. Hiding cliques for cryptographic security. *Designs, Codes and Cryptography*, 20(3):269–280, 2000.



- [53] Richard M Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.
- [54] Richard M Karp. Probabilistic analysis of some combinatorial search problems. In *Algorithms and Complexity: New Directions and Recent Results*. Academic Press, 1976.
- [55] Pascal Koiran and Anastasios Zouzias. Hidden cliques and the certification of the restricted isometry property. *IEEE Transactions on Information Theory*, 60(8):4999–5006, 2014.
- [56] Ludek Kucera. Expected complexity of graph partitioning problems. *Discrete Applied Mathematics*, 57(2-3):193–212, 1995.
- [57] Rio LaVigne, Andrea Lincoln, and Virginia Vassilevska Williams. Public-key cryptography in the fine-grained setting. In *Annual International Cryptology Conference*, pages 605–635. Springer, 2019.
- [58] Leonid A Levin. Average case complete problems. *SIAM Journal on Computing*, 15(1):285–286, 1986.
- [59] Richard J Lipton. New directions in testing. *Distributed Computing and Cryptography*, 2:191–202, 1989.
- [60] Zongming Ma and Yihong Wu. Computational barriers in minimax submatrix detection. *The Annals of Statistics*, 43(3):1089–1116, 2015.
- [61] Colin McDiarmid. Colouring random graphs. *Annals of Operations Research*, 1(3):183–200, 1984.
- [62] Frank McSherry. Spectral partitioning of random graphs. In *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, pages 529–537. IEEE, 2001.
- [63] David E Muller. Application of boolean algebra to switching circuit design and to error detection. *Transactions of the IRE professional group on electronic computers*, 3:6–12, 1954.
- [64] Jaroslav Nešetřil and Svatopluk Poljak. On the complexity of the subgraph problem. *Commentationes Mathematicae Universitatis Carolinae*, 26(2):415–419, 1985.
- [65] Gergely Palla, Imre Derényi, and Tamás Vicsek. The critical point of k-clique percolation in the erdős–rényi graph. *Journal of Statistical Physics*, 128(1-2):219–227, 2007.
- [66] B Pittel. On the probable behaviour of some algorithms for finding the stability number of a graph. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 92, pages 511–526. Cambridge University Press, 1982.

- [67] Mustazee Rahman and Balint Virag. Local algorithms for independent sets are half-optimal. *The Annals of Probability*, 45(3):1543–1577, 2017.
- [68] Alexander A Razborov. Lower bounds for the monotone complexity of some boolean functions. In *Soviet Math. Dokl.*, volume 31, pages 354–357, 1985.
- [69] Oded Regev. The learning with errors problem.
- [70] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):34, 2009.
- [71] Benjamin Rossman. On the constant-depth complexity of k-clique. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 721–730. ACM, 2008.
- [72] Benjamin Rossman. The monotone complexity of k-clique on random graphs. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 193–201. IEEE, 2010.
- [73] Benjamin Rossman. Lower bounds for subgraph isomorphism, 2016.
- [74] Madhu Sudan. Decoding of reed solomon codes beyond the error-correction bound. *Journal of complexity*, 13(1):180–193, 1997.
- [75] Leslie G Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, 8(3):410, 1979.
- [76] Van H Vu. A large deviation result on the number of small subgraphs of a random graph. *Combinatorics, Probability and Computing*, 10(1):79–94, 2001.
- [77] Virginia Vassilevska Williams. On some fine-grained questions in algorithms and complexity. In *Proceedings of the ICM*. World Scientific, 2018.
- [78] Raphael Yuster. Finding and counting cliques and independent sets in r-uniform hypergraphs. *Information Processing Letters*, 99(4):130–134, 2006.