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

Enric Boix Adserà
EECS, MIT
Cambridge, USA
eboix@mit.edu

Matthew Brennan
EECS, MIT
Cambridge, USA
brennanm@mit.edu

Guy Bresler
EECS, MIT
Cambridge, USA
guy@mit.edu

***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 prove the following:**

- ***Dense Erdős-Rényi hypergraphs*: Counting k-cliques on $G(n, c, s)$ with k and c constant matches its worst-case complexity up to a $\mathrm{polylog}(n)$ 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 hypergraphs*: When $c = \Theta(n^{-\alpha})$, for each fixed α 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 $\frac{\omega}{9} \le C \le 1$ and ω is the matrix multiplication constant. In the hypergraph case of $s \ge 3$, 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 known average-case hardness result on Erdos-Renyi hypergraphs based on a worst-case hardness assumption. We also analyze several natural algorithms for counting k-cliques in $G(n, c, s)$ that establish our upper bounds in the sparse case $c = \Theta(n^{-\alpha})$.

***Keywords*-average-case complexity; fine-grained complexity; worst-case-to-average-case reductions; graph algorithms; Erdős-Rényi hypergraphs**

I. Introduction

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. Our main result is 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. This 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 show that these average-case lower bounds often match algorithmic upper bounds.

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 $\epsilon > 0$ [1]. Natural polynomial time search algorithms and the Metropolis process find cliques of size approximately $\log_2 n$ but not $(1+\epsilon)\log_2 n$ [1], [2], [3], [4], [5]. 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 $2n\ln(d)/d$ [6], [7], [8]. In [9], it is shown that any algorithm probing $n^{2-\delta}$ edges of $G(n, 1/2)$ in ℓ 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 [10] and [3], 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 $k = o(\sqrt{n})$ [11], [12], [13], [14], [15], [16], [17], [18]. It has been shown that cliques of size $k = o(\sqrt{n})$ cannot be detected by the Metropolis process [3], low-degree sum of squares (SOS) relaxations [19] and statistical query algorithms [20]. 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 [21]. 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 [22], [23], [24], [25], [26], [27], [28], [29]. Recently, [30] also showed that super-polynomial length

2575-8454/19/$31.00 ©2019 IEEE
DOI 10.1109/FOCS.2019.00078

regular resolution is required to certify that Erdős-Rényi graphs do not contain cliques of size $k = o(n^{1/4})$.

Rossman [31], [32] has studied the classical k-CLIQUE decision problem on sparse Erdős-Rényi random graphs $G \sim G(n, c)$ at the critical threshold $c = \Theta\left(n^{-2/(k-1)}\right)$, 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 $v_1, v_2, \ldots, v_t$ such that v_{i+1} is a random common neighbor of $v_1, v_2, \ldots, v_i$ 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 $O\left(n^{k/4+O(1)}\right)$ 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 [31] and extended this lower bound to monotone circuits in [32]. A survey of this and related work can be found in [33].

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 coNP $\subseteq$ NP/poly [34], [35], [36]. 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. We now give an overview of our contributions.

A. 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 II-A.

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)[1] if $k = O(1)$ [37]. The best known worst-case algorithms up to subpolynomial factors are the $O\left(n^{\omega \lceil k/3 \rceil}\right)$ time algorithm of [38] 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.

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

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 $\mathrm{polylog}(n)$ queries to an average-case oracle that correctly counts k-cliques on a $1 - 1/\mathrm{polylog}(n)$ 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 $\tilde{\Omega}\left(n^{\omega \lfloor k/3 \rfloor}\right)$ 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 $\tilde{\Omega}\left(n^{\omega \lceil k/3 \rceil - \alpha \binom{k}{2}}\right)$ when $s = 2$ and $\tilde{\Omega}\left(n^{k - \alpha \binom{k}{s}}\right)$ when $s \geq 3$ for inputs drawn 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 $\mathrm{polylog}(n)$ factors.

Precise statements of our results can be found in Section II-B. 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 II-B. We discuss the necessity of this error tolerance and the multiplicative slowdown in our worst-case to average-case reduction in Section II-B. 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 I-B. The steps are described in detail in Section III.

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 [38] 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, τ and κ 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 $\alpha < 2\kappa^{-1} \leq \frac{2}{k-1}$. At $k = \tau + 1$, the first threshold becomes $\alpha < \frac{1}{k-1}$ which is exactly the k-clique percolation threshold [39], [40], [41]. 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 $G(n, c)$, [39] 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 [42]. Following the same heuristic as in [39], 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) **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, our upper and lower bounds approach each other for k small relative to $\kappa + 1$.
2) **Hypergraph Case** $(s \geq 3)$**.** 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 Section VI.

B. 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 [43], [34], [44], [45] that provides a method to efficiently compute a polynomial $P : \mathbb{F}^N \to \mathbb{F}$ of degree $d \leq |\mathbb{F}|/20$ on any worst-case input $x \in \mathbb{F}^N$, given an oracle $\tilde{P} : \mathbb{F}^N \to \mathbb{F}$ 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

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

This polynomial has been used in a number of papers, including by Goldreich and Rothblum [46] 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 $A \in \mathbb{F}_p^{\binom{n}{2}}$ [46] 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

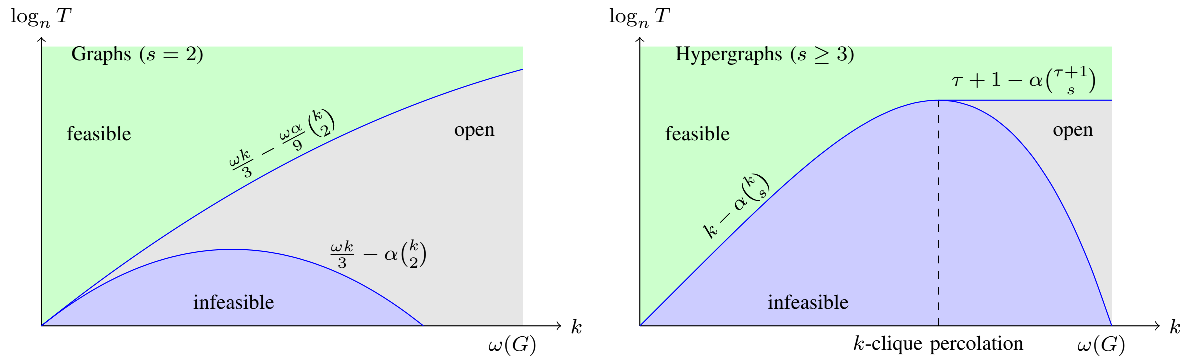

Figure 1. Comparison of our algorithms and average-case lower bounds for counting k-cliques in sparse Erdős-Rényi Hypergraphs $G(n, c, s)$ with $c = \Theta(n^{-\alpha})$. Green denotes runtimes T feasible for each k, blue denotes T infeasible given that the best known worst-case algorithms are optimal 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.

distribution, because gadgets inherently create non-random structure.

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

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

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 $X \sim \mathrm{Unif}\left[\mathbb{F}_p^{\binom{nk}{2}}\right]$ using an algorithm that evaluates P' on $G(n, c, 2)$ Erdős-Rényi graphs (i.e., inputs $Y \sim \mathrm{Ber}(c)^{\otimes \binom{nk}{2}}$)?

To this end we introduce a method for converting finite field elements to binary expansions: an efficient rejection sampling procedure to find $Y^{(0)}, \ldots, Y^{(t)}$ (for $t = \mathrm{poly}(c^{-1}(1-c)^{-1}\log(n))$) such that each $Y^{(i)}$ is close in total variation to $\mathrm{Ber}(c)^{\otimes \binom{nk}{2}}$, 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 Section IV, 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.

Now we algebraically manipulate P' as follows:

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

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 $\mathrm{poly}(c^{-1}(1-c)^{-1}\log(n))^{\binom{k}{2}}$ 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 Sections II-B and III.

C. Related Work on 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 $\mathsf{P}^{\#\mathsf{P}} = \mathsf{BPP}$ [47], [48]. (Under the slightly stronger assumption that $\mathsf{P}^{\#\mathsf{P}} \neq \mathsf{AM}$, and with different techniques, [49] proved that computing the permanent on large finite fields is hard even with exponentially small success probability.) Recently, [50] used the polynomial random self-reducibility result in the fine-grained setting in order to construct polynomials 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 [51] 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 [52] 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 [52], [53]. A good survey covering worst-case to average-case reductions in lattice cryptography is [54].

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 $\mathsf{coNP} \subseteq \mathsf{NP/poly}$ [34], [35], [36].

D. 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 $\mathrm{cl}_k(G)$ 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.

II. Problem Formulations and Average-Case Lower Bounds

A. 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 II.1. $\#(k, s)$-CLIQUE *denotes the problem of counting the number of k-cliques in an s-uniform hypergraph G.*

Definition II.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 II.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 [55] and $\#(k, s)$-CLIQUE is known to be #P-complete [56]. 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. As we show in Appendix A, DECIDE-(k, s)-CLIQUE also reduces to PARITY-(k, s)-CLIQUE with a multiplicative overhead of $O(k2^k)$ time.

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 [57]. 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 [58], [38]. 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 [59] 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 [60], [61]. In [62], 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 $f(k,s)\cdot \mathrm{poly}(n)$ – 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, [37] shows that solving DECIDE-$(k,2)$-CLIQUE in $n^{o(k)}$ time is ETH-hard for constant k[2]. We therefore conjecture that our k-clique problems take $n^{\Omega(k)}$ time on worst-case inputs when k is constant, as formalized below.

Conjecture II.4 (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 II.5 (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.*

Conjecture II.6 (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 II.6 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 II.7 (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]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.

B. 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 II.4, II.5 and II.7, imply average-case hardness on Erdős-Rényi hypergraphs for #(k,s)-CLIQUE and PARITY-(k,s)-CLIQUE.

Theorem II.8 (Worst-case to average-case reduction for #(k,s)-CLIQUE)**.** *There is an absolute constant $C>0$ such that if we define*

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

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 II.9 (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*

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

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 II.4 and II.7. 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 II.10 (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 II.4, 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 II.7, 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* $T(A,n) \geq \tilde{\Omega}\left(n^{\omega\lceil k/3\rceil}\right)$ *if $s=2$ and* $T(A,n) \geq \tilde{\Omega}(n^k)$ *if $s \geq 3$.*

Corollary II.11 (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 II.4, 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 $G(n,c,s)$ has runtime* $T(A,n) \geq n^{\Omega(k)}$.
2) *Assuming Conjecture II.7, 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 $G(n,c,s)$ has runtime* $T(A,n) \geq \tilde{\Omega}\left(n^{\omega\lceil k/3\rceil-\alpha\binom{k}{s}}\right)$ *if $s=2$ and* $T(A,n) \geq \tilde{\Omega}\left(n^{k-\alpha\binom{k}{s}}\right)$ *if $s \geq 3$.*

For PARITY-(k,s)-CLIQUE, we consider here the implications of Theorem II.9 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 II.12 (Average-case hardness of PARITY-(k,s)-CLIQUE on $G(n,1/2,s)$)**.** *Let k be constant. Assuming Conjecture II.5, 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 ϵ 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 V.1. Note that even when $\alpha > s\binom{k}{s-1}^{-1}$, GREEDY-RANDOM-SAMPLING and its derivative algorithms in Section V 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 Section V 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 II.7. Up to $\mathrm{polylog}(n)$ 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 II.7. 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 II.8 and II.9 require the error of the average-case blackbox on Erdős-Rényi hypergraphs goes 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 $1/\operatorname{polylog}(n)$ 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 $n^{-O(k^2 \log\log n)}$.

III. 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 II.8 and II.9. The reduction involves the following five main steps, the details of which are in Sections III-A to III-E.

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 III-F to complete the proofs of Theorems II.8 and II.9. 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 2 – 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 III.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 III.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 $\mathrm{Ber}(c)^{\otimes N}$.

A. 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.

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

$$E(G') = \left\{ \{(v_1, t_1), (v_2, t_2), \ldots, (v_s, t_s)\} : \{v_1, \ldots, v_s\} \in E(G) \text{ and } \substack{1 \le v_1 < v_2 < \cdots < v_s \le n \\ 1 \le t_1 < t_2 < \cdots < t_s \le k} \right\}.$$

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 $12\binom{k}{s} < p_1 < \cdots < p_T$ such that $\prod_{i=1}^T p_i > n^k$.

3) Define $L : (a,b) \in [n] \times [k] \mapsto b \in [k]$, and

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

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

(1) Use the procedure of [45] 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 $x_1, \ldots, x_M \sim \mathrm{Unif}[\mathbb{F}_{p_t}^N]$.

(2) For each $1 \le m \le M$, compute $P_{n,k,s}(x_m) \pmod{p_t}$ as follows:

(i) Use the rejection sampling procedure of Lemma IV.3 in order to sample $(Y^{(0)}, \ldots, Y^{(B)})$ close to $(\mathrm{Ber}(c)^{\otimes N})^{\otimes B}$ 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 $a : \binom{[k]}{s} \to \{0, \ldots, B\}$, 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 $\mathrm{Ber}(c)^{\otimes N}$. Use algorithm A and the recursive counting procedure of Lemma III.9 in order to compute $P_{n,k,s}(Y^{(a)})$ for each a.

(iii) Set $P_{n,k,s}(G') \leftarrow \sum_{a : \binom{[k]}{s} \to \{0,\ldots,B\}} 2^{|a|_1} \cdot P_{n,k,s}(Y^{(a)})$.

4) Since $0 \le P_{n,k,s}(G') \le n^k$, 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 2. 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.

Lemma III.3. *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 $T(B,n) \le T(A,n) + O(k^s n^s)$. 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

$$E(G') = \{\{(v_1,t_1),(v_2,t_2),\ldots,(v_s,t_s)\} \\ : \{v_1,\ldots,v_s\} \in E(G) \text{ and } \substack{1 \le v_1 < v_2 < \cdots < v_s \le n \\ 1 \le t_1 < t_2 < \cdots < t_s \le k}\Big\}$$

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

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

where $v_1 < v_2 < \cdots < v_k$. 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. ■

A corollary to Lemma III.3 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 II.4, II.5, and II.7 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.

B. Counting k-Cliques as a Low-Degree Polynomial

We now express the number of k-cliques of a k-partite hypergraph G with edge indicators $x \in \{0,1\}^N$ as a degree-

D polynomial $P_{n,k,s} : \{0,1\}^N \to \mathbb{Z}$ 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. 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 can be expressed as

$$P_{n,k,s}(x) = \sum_{\substack{\{u_1,\dots,u_k\} \subset V(G) \\ \forall i \;\; L(u_i)=i}} \prod_{\substack{S \subset [k] \\ |S|=s}} x_{u_S} \tag{1}$$

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 $P_{n,k,s,\mathbb{F}} : \mathbb{F}^N \to \mathbb{F}$. Observe that for any hypergraph $x \in \{0,1\}^N$, we have that

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

where $\mathrm{char}(\mathbb{F})$ 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 III.4. *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 $P_{n,k,s}(x) \le n^k$ 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 $P_{n,k,s,\mathbb{F}_{p_i}}(x) \equiv P_{n,k,s}(x) \pmod{p_i}$. ■

Proposition III.5. *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 $P_{n,k,s,\mathbb{F}}(x) \equiv P_{n,k,s}(x) \pmod{\mathrm{char}(\mathbb{F})}$. ■

C. 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 [45]. We follow the proof of [50] in order to present the lemma with explicit guarantees on the running time of the reduction.

Lemma III.6 (Theorem 4 of [45])**.** *Let $\mathbb{F}$ be a finite field with $|\mathbb{F}| = q$ elements. Let $N, D > 0$. Suppose $9 < D < q/12$. Let $f : \mathbb{F}^N \to \mathbb{F}$ be a polynomial of degree at most D. If there is an algorithm A running in time $T(A,N)$ such that*

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

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 $\mathbb{P}[B(x) = f(x)] > 2/3$.*

For completeness, we provide a proof of this lemma in Appendix B. Lemma III.6 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 $\mathbb{F}^N$.

D. 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 $x \sim \mathrm{Unif}\left[\mathbb{F}^N\right]$ to random hypergraphs, which correspond to $x \sim \mathrm{Unif}\left[\{0,1\}^N\right]$. 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 Unif$[\mathbb{F}_p]$ and related algorithmic corollaries from Section IV.

We first present the reduction that will be applied in the case of PARITY-(k,s)-CLIQUE. Given a map $a : \binom{[k]}{s} \to \{0,1,\dots,t-1\}$, let $a^* : [N] \to \{0,1,\dots,t-1\}$ denote the map induced by the labels $L : V(G) \to [k]$ of the vertices, when the indices of $[N]$ are identified with the possible k-partite hyperedges of G. Explicitly, if υ is a bijection between $[N]$ and the set of possible k-partite hyperedges of G under the labelling L, then define $a^*(i) = a(L(\upsilon(i)))$ 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 III.7. *Let p be prime and $t \ge 1$. 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 $y \sim$ Unif $\left[\mathbb{F}_p^N\right]$ 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 $x \sim$ Unif $\left[\mathbb{F}_{p^t}^N\right]$ in time $T(B,n) = O\left(Nt^4(\log p)^3 + t^D \cdot T(A,n)\right)$.*

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

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

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. [63]. Computing $x^{(0)}, \ldots, x^{(t-1)}$ then takes time $O(Nt^3(\log p)^3)$ because N applications of Gaussian elimination each take at most $O(t^3)$ operations over $\mathbb{F}_p$. [3] Note that since x is uniformly distributed and $\beta, \beta^p, \ldots, \beta^{p^{t-1}}$ form a basis, it follows that $x^{(0)}, x^{(1)}, \ldots, x^{(t-1)}$ are distributed i.i.d according to $\mathrm{Unif}\left[\mathbb{F}_p^N\right]$. For any map $b : [N] \to \{0, 1, \ldots, t-1\}$ 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.

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

As observed above, it holds that $x^{(a^*)} \sim \mathrm{Unif}\left[\mathbb{F}_p^N\right]$ for each a. Thus, computing $P_{n,k,s,\mathbb{F}}(x)$ 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. ■

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.

[3] For a good survey on normal bases, we recommend [64].

Lemma III.8. *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 $x \sim$ Unif $\left[\mathbb{F}_p^N\right]$ in time $T(B, n) = O\left(Npt\log(Np/\gamma) + t^D \cdot T(A, n)\right)$.*

Proof: We give a reduction computing $P_{n,k,s,\mathbb{F}_p}(x)$ where $x \sim \mathrm{Unif}\left[\mathbb{F}_p^N\right]$ given blackbox access to A. We first handle the case in which $p > 2$. For each $j \in [N]$, apply the algorithm from Lemma IV.3 to sample $x_j^{(0)}, x_j^{(1)}, \ldots, x_j^{(t-1)} \in \{0, 1\}$ satisfying

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

By Lemmas IV.2 and IV.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 $(x_j^{(0)}, \ldots, x_j^{(t-1)})$ with $(Z_j^{(0)}, \ldots, Z_j^{(t-1)}) \sim \mathrm{Ber}(c)^{\otimes k}$, so that $\mathbb{P}[x_j^{(i)} = Z_j^{(i)}\ \forall i, j] \ge 1-\gamma$. Moreover, we have $x_j^{(i)} \perp\!\!\!\perp x_l^{(k)}$ whenever $j \ne l$, so we may choose the $Z_j^{(i)}$ so that $Z_j^{(i)} \perp\!\!\!\perp Z_l^{(k)}$ whenever $j \ne l$.

As in the proof of Lemma III.7, given any map $b : [N] \to \{0, \ldots, t-1\}$, 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 $Z_1^{(b)}, \ldots, Z_N^{(b)}$ are independent and distributed as $\mathrm{Ber}(c)$. Therefore,

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

Now compute the following quantity, similarly to the calculations in Lemma III.7:

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

$$= \sum_{a:\binom{[k]}{s}\to\{0,\dots,t-1\}} \left(\prod_{S\in\binom{[k]}{s}} 2^{a(S)} \right) P_{n,k,s,\mathbb{F}_p}\left(Z^{(a^*)}\right).$$

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 IV.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 $d_{\mathrm{TV}}(\mathcal{L}(x_j^{(0)}, \dots, x_j^{(t-1)}), \mathrm{Ber}(c)^{\otimes t}) \leq \epsilon \triangleq \gamma/N$, and so that

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

By Lemma IV.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 $Z_j^{(i)} \sim \mathrm{Ber}(c)$ 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

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

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

E. 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 III.9. *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 $T(B, n) = O\left(2^k \cdot T(A, nk) + k^s n^s + k2^k\right)$ 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 $|\mathrm{cl}_k(H_S)|$ 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 $O\left(2^k \cdot T(A, nk)\right)$ time. From these counts $|\mathrm{cl}_k(H_S)|$, we now to inductively compute

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

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

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

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

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

Repeating the same proof over $\mathbb{F}_2$ yields an analogue of Lemma III.9 for PARITY-(k, s)-CLIQUE, as stated below.

Lemma III.10. *Lemma III.9 holds when* #(k, s)-CLIQUE *is replaced by* PARITY-(k, s)-CLIQUE.

F. Proofs of Theorems II.8 and II.9

We now combine Steps 1-5 formally in order to prove Theorems II.8 and II.9.

Proof of Theorem II.8: 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 $y \sim \mathrm{Unif}[\mathbb{F}_p^N]$ 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 δ 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 III.3 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 III.4 to construct A_0 from A_1 such that A_0 runs in time

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

The algorithm A_0 starts by using a sieve to find the first T primes $12\binom{k}{s} < p_1 < \cdots < p_T$ such that $\prod_{i=1}^T p_i > n^k$. Notice that $p_T \le 10 \log n^k$, 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 III.6 to construct A_1 from A_2 such that A_2 runs in time

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

4. Reduce to computing $P_{n,k,s}$ on random inputs in $\{0, 1\}^N$ We use Lemma III.8 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 δ 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 III.9 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 $4t^{\binom{k}{s}} \cdot 2^k \le \Upsilon_\#$. In this case, the error δ 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,

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

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

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

as $\binom{k}{s} \ge 3$ without loss of generality. ■

Proof of Theorem II.9: The proof of item 1 of Theorem II.9 is analogous to the proof of Theorem II.8, 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 $w \sim \mathrm{Unif}[\mathbb{F}_{2^\kappa}^N]$ 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 $y \sim \mathrm{Unif}[\mathbb{F}_2^N]$ with error probability $< \delta_2$. (The required value of δ_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 δ_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 III.3 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 III-B 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 III.6 to construct A_0 from A_1 such that A_0 runs in time

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

3. Reduce to computing $P_{n,k,s,\mathbb{F}_2}$ on random inputs in $\mathbb{F}_2^N$. We use Lemma III.7 to construct A_1 from A_2 such that A_1 runs in time

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

and has error probability at most $\delta_2 \cdot \kappa^{\binom{k}{s}}$ on random inputs $w \sim \mathrm{Unif}[\mathbb{F}_{2^\kappa}^N]$. Thus, A_2 must have error probability at most $\delta_2 < 1/(3\kappa^{\binom{k}{s}})$ on random inputs in $y \sim \mathrm{Unif}[\mathbb{F}_2^N]$ 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 III.8 to construct A_2 from A_3 such that A_2 runs in time

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

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 $\delta_2 \le 1/(3\kappa^{\binom{k}{s}})$ 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 $\delta_3 \le 1/(10(t\kappa)^{\binom{k}{s}})$, which is sufficient. For this choice of γ, 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 III.10 to construct A_3 from A such that A_3 runs in time

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

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

As in the theorem statement, let

$$\begin{aligned}\Upsilon_{P,1}(n,c,s,k) &\triangleq \left(C\cdot c^{-1}(1-c)^{-1}\right)^{\binom{k}{s}} \\ &\times \left((s\log k)\left(s\log n + \binom{k}{s}\log\log\binom{k}{s}\right)\right)^{\binom{k}{s}}\end{aligned}$$

for some large enough constant C.

If we take C large enough, then $(\kappa t)^{\binom{k}{s}} \le \frac{1}{10}\cdot 2^{-k}\cdot\Upsilon_{P,1}$, 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,

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

if we choose $C > 0$ large enough. Since $\binom{k}{s} \ge 3$ 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 $\delta_3 = \delta_2 \le 1/(3\kappa^{\binom{k}{s}})$. 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 $T(B,n) = O(\Upsilon_{P,2}\cdot(T(A,nk) + (nk)^s))$. ■

IV. 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.

Our argument uses finite Fourier analysis on $\mathbb{F}_p$. Given a function $f : \mathbb{F}_p \to \mathbb{R}$, define its Fourier transform to be $\hat{f} : \mathbb{F}_p \to \mathbb{C}$, 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 an simple claim showing that sufficiently long geometric progressions with ratio 2 in $\mathbb{F}_p$ contain a middle residue modulo p.

Claim IV.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 $1 \le i \le k-1$. Then if $k \ge 1 + \log_2(p/3)$, there is some j with $\frac{p}{3} \le a_j \le \frac{2p}{3}$.*

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 shows 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 $1 \le i \le k-1$ 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 $a'_k \ge 2^{k-1} \ge p/3$. Now consider the smallest j with $a'_j > p/3$. Then $p/3 \ge a'_{j-1} = a'_j/2$ by the minimality of i, and $p/3 \le a_j \le 2p/3$ 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. ■

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

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

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

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

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

Note that $|1 - q + q \cdot \omega^a| \le 1$ 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 $p/3 \le a \le 2p/3$ and $q \in [c, 1-c]$, then we have that

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

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

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

Fix some $s \in \mathbb{F}_p$ with $s \neq 0$. By Claim IV.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

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

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

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

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

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

$$\begin{aligned} d_{\mathrm{TV}}\left(\mathcal{L}(S), \mathrm{Unif}[\mathbb{F}_p]\right) &= \sum_{x \in \mathbb{F}_p} \left|p^{-1} - \mathbb{P}[S = x]\right|_+ \\ &= \frac{a(p-a)}{2^{t+1}p} \le \frac{p}{2^{t+1}} \end{aligned}$$

if $0 \le a \le p - 1$ is such that $2^{t+1} \equiv a \pmod p$. Therefore S is within total variation of $1/\mathrm{poly}(p)$ of $\mathrm{Unif}[\mathbb{F}_p]$ if $t = \Omega(\log p)$. However, note that for c constant and $\epsilon = 1/\mathrm{poly}(p)$, 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

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

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

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

where the second equality is due to the fact that the sequence 2^i has period r modulo p. Now observe that since $\frac{5}{9} + \frac{4}{9} \cdot \cos(x) \ge e^{-x^2}$, we have that

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

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 IV.3. *Let $p > 2$ be prime. Suppose that $c \le q_1, q_2, \dots, q_t \le 1 - c$ for some $c \in (0, 1/2]$ and that $Z_i \sim \mathrm{Ber}(q_i)$ 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 $R \sim \mathrm{Unif}[\mathbb{F}_p]$. If $S = Y \pmod p$ is as in the previous lemma and $\Delta = d_{TV}\left(\mathcal{L}(S), \mathrm{Unif}[\mathbb{F}_p]\right) < p^{-1}$, then it holds that*

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

Proof: Note that the $x \to Y_x$ defines a Markov transition sending $S \to Y$ and $R \to Y_R$. The data-processing inequality yields $d_{\mathrm{TV}}(\mathcal{L}(Y), \mathcal{L}(Y_R)) \le d_{\mathrm{TV}}(\mathcal{L}(S), \mathcal{L}(R)) = \Delta$, 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 mth 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

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

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

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

which completes the proof of the second item. ∎

We conclude this section with a sampling result similar to Lemma IV.3, but for the $p = 2$ case.

Lemma IV.4. *Let $R \sim \mathrm{Unif}[\mathbb{F}_2]$, 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 $d_{TV}(\mathcal{L}(X), \mathrm{Ber}(c)^{\otimes t}) < \epsilon$.*

Proof: Let $Z_1, \dots, Z_t \overset{i.i.d}{\sim} \mathrm{Ber}(c)$. 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 $d_{\mathrm{TV}}(\mathcal{L}(\sum_{i=1}^t Z_i), \mathcal{L}(R)) \leq \epsilon/8$. Sample the distribution

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

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 $Z \sim \mathrm{Ber}(c)^{\otimes t}$ and checking if $\sum_{i=1}^t Z_i = R$. By triangle inequality, it suffices to show that $d_{\mathrm{TV}}(\mathcal{L}(X), \mathrm{Ber}(c)^{\otimes t}) \leq \epsilon/2$. This is true because for any $\omega \in \{0,1\}^t$,

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

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

V. 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 II-B, 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.

A. 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

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

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 $\mathrm{CN}_G(v_1, v_2, \dots, v_i)$ 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 [1], [2], the Metropolis process examined in [3] and the greedy procedure solving k-CLIQUE on $G(n,c)$ with $c = \Theta\left(n^{-2/(k-1)}\right)$ discussed by Rossman in [33]. 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 [4], [5]. 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 τ 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. Its proof is included in Appendix C for completeness.

Lemma V.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 $\mathbb{E}[|\mathrm{cl}_k(G)|] = \binom{n}{k}c^{\binom{k}{s}}$ 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 positive integer t.*

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.

B. 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 $S = \mathrm{cl}_k(G)$. 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 $O(\log|\mathrm{cl}_k(G)|) = O(k\log n)$ 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 $S = \mathrm{cl}_k(G)$ and therefore solve $\#(k,s)$-CLIQUE.

Theorem V.2. *Let k and s be constants and $c = \Theta(n^{-\alpha})$ for some $\alpha \in (0,1)$. Let τ 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 $S = \mathrm{cl}_k(G)$ 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

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

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

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

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 $V(G)\backslash\{v_1, v_2, \dots, v_k\}$. Note that $Y_{v,i} \sim \mathrm{Bin}\left(n-k, c^{\binom{i}{s-1}}\right)$ and that $|\mathrm{CN}_G(v_1, v_2, \dots, v_i)| = k - i + Y_{v,i}$. The standard Chernoff bound for the binomial distribution implies that for all $\delta_i > 0$,

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

Now define κ_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 δ_i ensure that the Chernoff upper bounds above are each at most $\exp\left(-\frac{1}{3}(\log n)^{1+\epsilon}\right)$ for each i. A union bound implies that with probability at least $1 - k\exp\left(-\frac{1}{3}(\log n)^{1+\epsilon}\right)$, it holds that

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

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

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

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

$$\begin{aligned}&\mathbb{P}\Big[Z_v \geq 2n^{\tau+1}c^{\binom{\tau+1}{s}}(\log n)^{3(k-1-\tau)(1+\epsilon)} \\ &\qquad\qquad \Big| \{v_1, v_2, \dots, v_k\} \in \mathrm{cl}_k(G)\Big] \\ &\quad \leq k\exp\left(-\frac{1}{3}(\log n)^{1+\epsilon}\right) = n^{-\omega(1)}\end{aligned}$$

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

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

Note that $\overline{B} = \bigcup_{k\text{-tuples } v} A_v$ and thus a union bound implies that $\mathbb{P}[B] \ge 1 - \sum_v \mathbb{P}[A_v] \ge 1 - n^k \cdot n^{-\omega(1)} = 1 - n^{-\omega(1)}$ 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 $S = \mathrm{cl}_k(G)$ 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

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

since T is chosen so that $T \ge Z_v (\log n)^{3(1+\epsilon)}$ 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, $S = \mathrm{cl}_k(G)$ after the T rounds of GREEDY-RANDOM-SAMPLING. This completes the proof of the theorem in the case $k \ge \tau + 1$.

We now handle the case $k < \tau + 1$ through a nearly identical argument. Define κ_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 $i \le k - 1 < \tau$. Define the event

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

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

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 V.3. *Suppose that k and s are constants and $c = \Theta(n^{-\alpha})$ for some $\alpha \in (0,1)$. Let τ be the largest integer satisfying $\alpha\binom{\tau}{s-1} < 1$. Then it follows that*

1) *If $k \ge \tau + 1$, 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 V.1, the hypergraph $G \sim G(n,c,s)$ has clique number $\omega(G) \le \kappa + 2$ with probability $1 - 1/\mathrm{poly}(n)$ if where $\kappa \ge s$ 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 $1 - 1/\mathrm{poly}(n)$ 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 $0 < \alpha \le s\binom{k-1}{s-1}^{-1}$. In the graph case of $s = 2$, this simplifies to the familiar condition that $0 < \alpha \le \frac{2}{k-1}$. 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 $G \sim G(n,c,s)$.

C. 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 $O\left(n^{\omega\lfloor k/3 \rfloor + (k \pmod 3)}\right)$ time worst-case algorithm for $\#(k,2)$-CLIQUE [38]. Our improved runtime for GREEDY-RANDOM-SAMPLING is stated in the following theorem.

Theorem V.4. *Suppose that $k > 2$ is a fixed positive integer and $c = \Theta(n^{-\alpha})$ where $0 < \alpha \le \frac{2}{k-1}$ 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 $S_1 = \mathrm{cl}_{\lfloor k/3\rfloor}(G)$ and $S_2 = \mathrm{cl}_{\lceil k/3\rceil}(G)$ with the number of iterations T as given in Theorem V.2.
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
$$M_P = \begin{cases} M_1^2 & \text{if } k \equiv 0 \pmod 3 \\ M_1M_2 & \text{if } k \equiv 1 \pmod 3 \\ M_2M_3 & \text{if } k \equiv 2 \pmod 3 \end{cases}$$
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 V.2, 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 $\mathcal{C} \in \mathrm{cl}_k(G)$ 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 $\alpha \le \frac{2}{k-1}$, then $\tau \ge \lfloor\frac{k}{2}\rfloor - 1$. If $k \ge 3$, then it follows that $\tau + 1 \ge \lfloor\frac{k}{2}\rfloor \ge \lceil\frac{k}{3}\rceil$. It follows by Theorem V.2 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 $\tilde{O}\left(\max\{|S_1|,|S_2|\}^\omega\right) = \tilde{O}\left(n^{\omega\lceil k/3\rceil+\omega-\omega\alpha\binom{\lceil k/3\rceil}{2}}\right)$ 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 $\omega \ge 2$. ■

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 $k \ne 0 \pmod 3$. Note that the proof above implicitly used a weak large deviations bound on $|\mathrm{cl}_k(G)|$. More precisely, it used the fact that if GREEDY-RANDOM-SAMPLING with T iterations succeeds, then $|\mathrm{cl}_k(G)| \le T$. Theorem V.2 thus implies that $|\mathrm{cl}_k(G)|$ 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 $k \le \tau + 1$, these upper bounds are a polylog(n) factor from the expectation of $|\mathrm{cl}_k(G)|$. The upper tails of $|\mathrm{cl}_k(G)|$ and more generally of the counts of small subhypergraphs in $G(n,c,s)$ have been studied extensively in the literature. We refer to [65], [66], [67], [68] 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
$$M_H(n,c) = \max\left\{m \le \binom{n}{s} : \forall H' \subseteq H \text{ it holds that } N(n,m,H') \le n^{|V(H')|}c^{|E(H')|}\right\}$$
The following large deviations result from [69] generalizes a graph large deviations bound from [67] to hypergraphs to obtain the following result.

Theorem V.5 (Theorem 4.1 from [69])**.** *For every s-uniform hypergraph H and every fixed $\epsilon > 0$, there is a constant $C(\epsilon,H)$ such that for all $n \ge |V(H)|$ and $c \in (0,1)$, it holds that*
$$\mathbb{P}\left[X_H \ge (1+\epsilon)\mathbb{E}[X_H]\right] \le \exp\left(-C(\epsilon,H)\cdot M_H(n,c)\right)$$
where X_H is the number of copies of H in $G \sim G(n,c,s)$.

Proposition 4.3 in [69] shows that if H is a d-regular s-uniform hypergraph and $c \ge n^{-s/d}$ then $M_H(n,c) = \Theta(n^s c^d)$. This implies that
$$\mathbb{P}\left[|\mathrm{cl}_k(G)| \ge (1+\epsilon)\binom{n}{k}c^{\binom{k}{s}}\right] \le \exp\left(-C'(\epsilon)\cdot n^s c^{\binom{k-1}{s-1}}\right)$$

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

D. 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} as 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 V.5. Then we have the following guarantees for the algorithm IT-GEN-CLIQUES.

Theorem V.6. *Suppose that k and s are constants and $c = \Theta(n^{-\alpha})$ for some $\alpha \in (0, 1)$. Let τ 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 $S_k = \mathrm{cl}_k(G)$ 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 $S_k = \mathrm{cl}_k(G)$ with probability $1 - n^{-\omega(1)}$ in IT-GEN-CLIQUES. By a union bound and Theorem V.5, it follows that $|\mathrm{cl}_t(G)| < C_t$ 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 $S_t = \mathrm{cl}_t(G)$ for each $s-1 \leq t \leq k$ conditioned on this event. Note that $\mathrm{cl}_{s-1}(G)$ is by definition the set of all $(s-1)$-subsets of $[n]$ and thus $S_{s-1} = \mathrm{cl}_{s-1}(G)$. If $S_t = \mathrm{cl}_t(G)$, 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 $A = \mathcal{C}\backslash\{v\} \in \mathrm{cl}_t(G)$ 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. ■

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 $\mathrm{polylog}(n)$ 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 V.4, 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 $G(n, c)$.

VI. Extensions and Open Problems

In this section, 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 II.8 and II.9 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 e, 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 σ, one would have to choose $e \ll \sigma$ 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 II.8 and II.9 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 $c(e) \in [c, 1-c]$.

General Subgraph Counts: Let H be a hypergraph on k vertices. Let H-COUNTING be the problem of counting the number of occurrences (as an induced subgraph) of H in a hypergraph G. Can one show that H-COUNTING in the worst case reduces to H-COUNTING in the average case on Erdős-Rényi hypergraphs?

Our reduction (Theorem II.8) applies to the special case when H is a clique. Unfortunately, the proof of Theorem II.8 breaks down when counting general hypergraphs. First, the reductions to and from k-partite hypergraphs (steps 1 and 5) no longer work, because H contains non-edges, and therefore there may be a copy of H that contains more than one vertex in a given k-partition. In order to remedy this, we could consider the modification H-COUNTING$'$ of the H-COUNTING problem that respects k-partite structure, by only counting the copies of H in a k-partite hypergraph G, such that the k vertices of the copy of H lie in the k different parts of the vertex partition of G. For this modified problem, the strategy of our reduction still fails – this time at Step 4, because the polynomial that counts copies of H in G is not homogeneous. Indeed, for clique-counting, Step 4 of the reduction uses the fact that the variables of the clique-counting polynomial can be split up into $\binom{k}{s}$ groups, such that each monomial contained exactly one variable from each group.

REFERENCES

[1] R. M. Karp, "Probabilistic analysis of some combinatorial search problems," in *Algorithms and Complexity: New Directions and Recent Results*. Academic Press, 1976.

[2] G. R. Grimmett and C. J. McDiarmid, "On colouring random graphs," in *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 77, no. 2. Cambridge University Press, 1975, pp. 313–324.

[3] M. Jerrum, "Large cliques elude the metropolis process," *Random Structures & Algorithms*, vol. 3, no. 4, pp. 347–359, 1992.

[4] C. McDiarmid, "Colouring random graphs," *Annals of Operations Research*, vol. 1, no. 3, pp. 183–200, 1984.

[5] 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*, vol. 92, no. 3. Cambridge University Press, 1982, pp. 511–526.

[6] D. Gamarnik and M. Sudan, "Limits of local algorithms over sparse random graphs," in *Proceedings of the 5th conference on Innovations in theoretical computer science*. ACM, 2014, pp. 369–376.

[7] A. Coja-Oghlan and C. Efthymiou, "On independent sets in random graphs," *Random Structures & Algorithms*, vol. 47, no. 3, pp. 436–486, 2015.

[8] M. Rahman and B. Virag, "Local algorithms for independent sets are half-optimal," *The Annals of Probability*, vol. 45, no. 3, pp. 1543–1577, 2017.

[9] U. Feige, D. Gamarnik, J. Neeman, M. Z. Rácz, and P. Tetali, "Finding cliques using few probes," *arXiv preprint arXiv:1809.06950*, 2018.

[10] L. Kucera, "Expected complexity of graph partitioning problems," *Discrete Applied Mathematics*, vol. 57, no. 2-3, pp. 193–212, 1995.

[11] N. Alon, M. Krivelevich, and B. Sudakov, "Finding a large hidden clique in a random graph," *Random Structures and Algorithms*, vol. 13, no. 3-4, pp. 457–466, 1998.

[12] U. Feige and R. Krauthgamer, "Finding and certifying a large hidden clique in a semirandom graph," *Random Structures and Algorithms*, vol. 16, no. 2, pp. 195–208, 2000.

[13] F. McSherry, "Spectral partitioning of random graphs," in *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*. IEEE, 2001, pp. 529–537.

[14] U. Feige and D. Ron, "Finding hidden cliques in linear time," in *21st International Meeting on Probabilistic, Combinatorial, and Asymptotic Methods in the Analysis of Algorithms (AofA'10)*. Discrete Mathematics and Theoretical Computer Science, 2010, pp. 189–204.

[15] B. P. Ames and S. A. Vavasis, "Nuclear norm minimization for the planted clique and biclique problems," *Mathematical programming*, vol. 129, no. 1, pp. 69–89, 2011.

[16] Y. Dekel, O. Gurel-Gurevich, and Y. Peres, "Finding hidden cliques in linear time with high probability," *Combinatorics, Probability and Computing*, vol. 23, no. 1, pp. 29–49, 2014.

[17] Y. Deshpande and A. Montanari, "Finding hidden cliques of size $\sqrt{N/e}$ in nearly linear time," *Foundations of Computational Mathematics*, vol. 15, no. 4, pp. 1069–1128, 2015.

[18] Y. Chen and J. Xu, "Statistical-computational tradeoffs in planted problems and submatrix localization with a growing number of clusters and submatrices," *Journal of Machine Learning Research*, vol. 17, no. 27, pp. 1–57, 2016.

[19] B. Barak, S. B. Hopkins, J. Kelner, P. Kothari, A. Moitra, and A. 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*. IEEE, 2016, pp. 428–437.

[20] V. Feldman, E. Grigorescu, L. Reyzin, S. Vempala, and Y. Xiao, "Statistical algorithms and a lower bound for detecting planted cliques," in *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*. ACM, 2013, pp. 655–664.

[21] A. Juels and M. Peinado, "Hiding cliques for cryptographic security," *Designs, Codes and Cryptography*, vol. 20, no. 3, pp. 269–280, 2000.

[22] Q. Berthet and P. Rigollet, "Complexity theoretic lower bounds for sparse principal component detection." in *COLT*, 2013, pp. 1046–1066.

[23] P. Koiran and A. Zouzias, "Hidden cliques and the certification of the restricted isometry property," *IEEE Transactions on Information Theory*, vol. 60, no. 8, pp. 4999–5006, 2014.

[24] Y. Chen, "Incoherence-optimal matrix completion," *IEEE Transactions on Information Theory*, vol. 61, no. 5, pp. 2909–

2923, 2015.

[25] B. E. Hajek, Y. Wu, and J. Xu, "Computational lower bounds for community detection on random graphs." in *COLT*, 2015, pp. 899–928.

[26] Z. Ma and Y. Wu, "Computational barriers in minimax submatrix detection," *The Annals of Statistics*, vol. 43, no. 3, pp. 1089–1116, 2015.

[27] M. Brennan, G. Bresler, and W. Huleihel, "Reducibility and computational lower bounds for problems with planted sparse structure," in *Conference On Learning Theory*, 2018, pp. 48–166.

[28] ——, "Universality of computational lower bounds for submatrix detection," *arXiv preprint arXiv:1902.06916*, 2019.

[29] M. Brennan and G. Bresler, "Optimal average-case reductions to sparse pca: From weak assumptions to strong hardness," *arXiv preprint arXiv:1902.07380*, 2019.

[30] A. Atserias, I. Bonacina, S. F. de Rezende, M. Lauria, J. Nordström, and A. Razborov, "Clique is hard on average for regular resolution," in *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*. ACM, 2018, pp. 866–877.

[31] B. Rossman, "On the constant-depth complexity of k-clique," in *Proceedings of the fortieth annual ACM symposium on Theory of computing*. ACM, 2008, pp. 721–730.

[32] ——, "The monotone complexity of k-clique on random graphs," in *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*. IEEE, 2010, pp. 193–201.

[33] ——, "Lower bounds for subgraph isomorphism," 2016.

[34] J. Feigenbaum and L. Fortnow, "Random-self-reducibility of complete sets," *SIAM Journal on Computing*, vol. 22, no. 5, pp. 994–1005, 1993.

[35] A. Bogdanov and L. Trevisan, "On worst-case to average-case reductions for np problems," *SIAM Journal on Computing*, vol. 36, no. 4, pp. 1119–1159, 2006.

[36] ——, "Average-case complexity," *Foundations and Trends® in Theoretical Computer Science*, vol. 2, no. 1, pp. 1–106, 2006.

[37] J. Chen, X. Huang, I. A. Kanj, and G. Xia, "Strong computational lower bounds via parameterized complexity," *Journal of Computer and System Sciences*, vol. 72, no. 8, pp. 1346–1367, 2006.

[38] J. Nesetril and S. Poljak, "On the complexity of the subgraph problem," *Commentationes Mathematicae Universitatis Carolinae*, vol. 26, no. 2, pp. 415–419, 1985.

[39] I. Derényi, G. Palla, and T. Vicsek, "Clique percolation in random networks," *Physical review letters*, vol. 94, no. 16, p. 160202, 2005.

[40] G. Palla, I. Derényi, and T. Vicsek, "The critical point of k-clique percolation in the erdős–rényi graph," *Journal of Statistical Physics*, vol. 128, no. 1-2, pp. 219–227, 2007.

[41] S. N. Dorogovtsev, A. V. Goltsev, and J. F. Mendes, "Critical phenomena in complex networks," *Reviews of Modern Physics*, vol. 80, no. 4, p. 1275, 2008.

[42] B. Bollobás and O. Riordan, "Clique percolation," *Random Structures & Algorithms*, vol. 35, no. 3, pp. 294–322, 2009.

[43] R. J. Lipton, "New directions in testing." *Distributed Computing and Cryptography*, vol. 2, pp. 191–202, 1989.

[44] P. Gemmell, R. Lipton, R. Rubinfeld, M. Sudan, and A. Wigderson, "Self-testing/correcting for polynomials and for approximate functions," in *Proceedings of the twenty-third annual ACM symposium on Theory of computing*. ACM, 1991, pp. 33–42.

[45] P. Gemmell and M. Sudan, "Highly resilient correctors for polynomials," *Inf. Process. Lett.*, vol. 43, no. 4, pp. 169–174, 1992.

[46] O. Goldreich and G. 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)*. IEEE, 2018, pp. 77–88.

[47] M. Sudan, "Decoding of reed solomon codes beyond the error-correction bound," *Journal of complexity*, vol. 13, no. 1, pp. 180–193, 1997.

[48] J.-Y. Cai, A. Pavan, and D. Sivakumar, "On the hardness of permanent," in *Annual Symposium on Theoretical Aspects of Computer Science*. Springer, 1999, pp. 90–99.

[49] U. Feige and C. Lund, "On the hardness of computing the permanent of random matrices," in *Proceedings of the twenty-fourth annual ACM symposium on Theory of computing*. ACM, 1992, pp. 643–654.

[50] M. Ball, A. Rosen, M. Sabin, and P. N. Vasudevan, "Average-case fine-grained hardness," in *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*. ACM, 2017, pp. 483–496.

[51] D. Gamarnik, "Computing the partition function of the sherrington-kirkpatrick model is hard on average," *arXiv preprint arXiv:1810.05907*, 2018.

[52] M. Ajtai, "Generating hard instances of lattice problems," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. ACM, 1996, pp. 99–108.

[53] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," *Journal of the ACM (JACM)*, vol. 56, no. 6, p. 34, 2009.

[54] ——, "The learning with errors problem."

[55] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of computer computations*. Springer, 1972, pp. 85–103.

[56] L. G. Valiant, "The complexity of enumeration and reliability problems," *SIAM Journal on Computing*, vol. 8, no. 3, p. 410, 1979.

[57] R. Yuster, "Finding and counting cliques and independent sets in r-uniform hypergraphs," *Information Processing Letters*, vol. 99, no. 4, pp. 130–134, 2006.

[58] A. Itai and M. Rodeh, "Finding a minimum circuit in a graph," *SIAM Journal on Computing*, vol. 7, no. 4, pp. 413–423, 1978.

[59] A. A. Razborov, "Lower bounds for the monotone complexity of some boolean functions," in *Soviet Math. Dokl.*, vol. 31, 1985, pp. 354–357.

[60] N. Alon and R. B. Boppana, "The monotone circuit complexity of boolean functions," *Combinatorica*, vol. 7, no. 1, pp. 1–22, 1987.

[61] K. Amano and A. 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*, vol. 35, no. 1, pp. 201–216, 2005.

[62] R. G. Downey and M. R. Fellows, "Fixed-parameter tractability and completeness ii: On completeness for w [1]," *Theoretical Computer Science*, vol. 141, no. 1-2, pp. 109–131, 1995.

[63] E. Bach, J. Driscoll, and J. Shallit, "Factor refinement," *Journal of Algorithms*, vol. 15, no. 2, pp. 199–222, 1993.

[64] S. Gao, "Normal bases over finite fields," *Doctoral thesis, Waterloo*, 1993.

[65] V. H. Vu, "A large deviation result on the number of small subgraphs of a random graph," *Combinatorics, Probability and Computing*, vol. 10, no. 1, pp. 79–94, 2001.

[66] S. Janson and A. Ruciński, "The infamous upper tail," *Random Structures & Algorithms*, vol. 20, no. 3, pp. 317–342, 2002.

[67] S. Janson, K. Oleszkiewicz, and A. Ruciński, "Upper tails

for subgraph counts in random graphs," *Israel Journal of Mathematics*, vol. 142, no. 1, pp. 61–92, 2004.
[68] R. DeMarco and J. Kahn, "Tight upper tail bounds for cliques," *Random Structures & Algorithms*, vol. 41, no. 4, pp. 469–487, 2012.
[69] A. Dudek, J. Polcyn, and A. Ruciński, "Subhypergraph counts in extremal and random hypergraphs and the fractional q-independence," *Journal of combinatorial optimization*, vol. 19, no. 2, pp. 184–199, 2010.
[70] D. E. Muller, "Application of boolean algebra to switching circuit design and to error detection," *Transactions of the IRE professional group on electronic computers*, vol. 3, pp. 6–12, 1954.

APPENDIX

A. Reduction from DECIDE-(k,s)-CLIQUE to PARITY-(k,s)-CLIQUE

The following is a precise statement and proof of the reduction from DECIDE-(k,s)-CLIQUE to PARITY-(k,s)-CLIQUE claimed in Section II-A.

Lemma A.1. *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 $O(k2^k|A|)$ and solves* DECIDE-(k,s)-CLIQUE *with error $< 1/3$ on any s-uniform hypergraph G.*

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

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

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, then it is nonzero on at least a 2^{-k} fraction of inputs. One way to see this is that if we evaluate P_G at all points $a \in \{0,1\}^m$, the result is a non-zero Reed-Muller codeword in $RM(k,m)$. Since the distance of the $RM(k,m)$ code is 2^{m-k}, and the block-length is 2^m, the claim follows [70]. 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. ∎

B. Proof of Lemma III.6

We restate and prove Lemma III.6.

Lemma A.2 (Theorem 4 of [45])**.** *Let $\mathbb{F}$ be a finite field with $|\mathbb{F}| = q$ elements. Let $N, D > 0$. Suppose $9 < D < q/12$. Let $f : \mathbb{F}^N \to \mathbb{F}$ be a polynomial of degree at most D. If there is an algorithm A running in time $T(A,N)$ such that*

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

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 $x \in \mathbb{F}^N$,

$$\mathbb{P}[B(x) = f(x)] > 2/3.$$

Proof: Our proof of the lemma is based off of the proof that appears in [50]. The only difference is that in [50], 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 $y_1, y_2 \overset{i.i.d}{\sim} \mathrm{Unif}[\mathbb{F}^N]$, 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 $m = 12D$. ∎

C. Clique Counts in Sparse Erdős-Rényi Hypergraphs

We prove the following classical lemma from Section V-A.

Lemma A.3. *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 $\mathbb{E}[|\mathrm{cl}_k(G)|] = \binom{n}{k} c^{\binom{k}{s}}$ 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 positive 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 $\mathbb{E}[|\mathrm{cl}_k(G)|] = \binom{n}{k} c^{\binom{k}{s}}$. Now consider taking $k = \kappa + 2 + t$ and note that

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

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 $\mathbb{P}[\omega(G) \geq k] = \mathbb{P}[|\mathrm{cl}_k(G)| \geq 1] \leq \mathbb{E}[|\mathrm{cl}_k(G)|]$, completing the proof of the lemma. ■