SIAM J. COMPUT.
Vol. 0, No. 0, pp. FOCS19-39–FOCS19-80

THE AVERAGE-CASE COMPLEXITY OF COUNTING CLIQUES IN ERDŐS–RÉNYI HYPERGRAPHS*

ENRIC BOIX-ADSERÀ†, MATTHEW BRENNAN‡, AND GUY BRESLER†

Dedicated to the memory of our dear colleague and friend, Matthew Brennan

Abstract. 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: (1) Dense Erdős–Rényi graphs and hypergraphs: Counting k-cliques on $G(n, c, s)$ with k and c constant matches its worst-case complexity up to a polylog(n) factor. Assuming randomized ETH, it takes $n^{\Omega(k)}$ time to count k-cliques in $G(n, c, s)$ if k and c are constant. (2) Sparse Erdős–Rényi graphs and hypergraphs: When $c = \Theta(n^{-\alpha})$, we give several algorithms exploiting the sparsity of $G(n, c, s)$ that are faster than the best known worst-case algorithms. Complementing this, based on a fine-grained worst-case assumption, our reduction implies a different average-case phase diagram for each fixed α depicting a tradeoff between a runtime lower bound and k. Surprisingly, in the hypergraph case ($s \geq 3$), these lower bounds are tight against our algorithms exactly when c is above the Erdős–Rényi k-clique percolation threshold. Our reduction yields the first known average-case hardness result on Erdős–Rényi hypergraphs based on worst-case hardness conjectures. We also give a variant of our worst-case to average-case reduction for computing the parity of the k-clique count that requires a milder assumption on the error probability of the blackbox solving the problem on $G(n, c, s)$.

Key words. average-case complexity, fine-grained complexity, worst-case-to-average-case reductions, graph algorithms, random graphs

AMS subject classifications. 68Q17, 68Q87, 60C05

DOI. 10.1137/20M1316044

1. 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 reduction for counting k-cliques on worst-case hypergraphs given a blackbox algorithm solving the problem on $G(n, c, s)$ with low error probability. Our approach is closely related to the recent work [43], which showed a worst-case to average-case reduction for counting cliques for a particular efficiently samplable distribution on graphs. Our reduction yields two different sets of average-case lower bounds for counting k-cliques in graphs sampled from the natural distribution $G(n, c, s)$ in the dense and sparse cases of $c = \Theta(1)$ and $c = \Theta(n^{-\alpha})$, with tradeoffs between runtime and c. 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. A body of work has analyzed algorithms for finding large cliques in Erdős–Rényi graphs[1] [56, 3, 32, 60, 34, 5, 24, 27, 21], and hardness results have been shown for greedy algorithms [54, 45, 51, 59, 64], local algorithms [38, 22, 65], query models [31],

*Received by the editors January 30, 2020; accepted for publication (in revised form) June 7, 2021; published electronically September 14, 2021.
https://doi.org/10.1137/20M1316044

†Department of EECS, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (eboix@mit.edu, guy@mit.edu).

‡The author is deceased. Former address: Department of EECS, Massachusetts Institute of Technology, Cambridge, MA 02139 USA.

[1] in both ordinary Erdős–Rényi graphs and the planted clique model.

bounded-depth circuits [69], monotone circuits [70], low-degree sum of squares (SOS) relaxations [9], statistical query algorithms [36], and resolution [6]. The hardness of clique problems on Erdős–Rényi graphs has been used as an average-case assumption in cryptography [52] and to show information-computation gaps in a variety of statistical problems [10, 55, 20, 46, 58, 15, 16, 14].

All of the above lower bounds for clique problems on Erdős–Rényi random graphs are against restricted classes of 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 [35, 12, 11]. 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 natural problem of counting k-cliques in s-uniform Erdős–Rényi hypergraphs $G(n, c, s)$ with edge density c.

A motivating recent work by Goldreich and Rothblum [43] also considered worst-case to average-case reductions for k-clique counting. They provided such a reduction mapping to an efficiently samplable distribution on graphs with a high min-entropy of $\tilde{\Omega}(n^2)$. In contrast to [43], our objectives are to (1) map precisely to the natural distribution $G(n, c, s)$ for different edge densities c, including $c = \Theta(1)$ and the sparse case $c = \Theta(n^{-\alpha})$; and (2) to characterize the tradeoff between the time-complexity of counting k-cliques in $G(n, c, s)$ and the sparsity parameter α. Achieving this requires new ingredients for the self-reducibility of counting k-cliques as a low-degree polynomial and a tight analysis of random biased binary expansions over $\mathbb{F}_p$ with finite Fourier analysis.

However, our techniques also come at the cost of requiring a low error probability ($1/\text{polylog}(n)$ in the dense case and $1/\text{poly}(n)$ in the sparse case) for the average-case blackbox solving k-clique counting on $G(n, c, s)$. This is in contrast to [43], where a very high error probability of $1 - 1/\text{polylog}(n)$ is tolerated. It remains an interesting open problem to extend our results for $G(n, c, s)$ to tolerate higher error blackboxes. This error tolerance and open problem are discussed further in sections 2.2 and 6, and how our techniques relate to those in [43] is discussed in sections 1.2 and 3. As a step towards increasing the allowed blackbox error, we also give a variant of our reduction for computing the *parity* of the k-clique count that only requires a constant bound on the error probability (for each fixed k) of the blackbox algorithm solving the problem on $G(n, c, s)$ when $c = 1/2$. We now give an overview of our contributions.

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

Worst-case to average-case reduction. We give a worst-case to average-case reduction from counting k-cliques in worst-case s-uniform hypergraphs to counting k-cliques in hypergraphs drawn from $G(n, c, s)$. The key guarantees of this reduction are summarized in the following simplified version of our main theorem.

THEOREM 1.1 (simplified main result). *If $2 \le s \le k$ are constant integers and $c = c(n)$ satisfies $0 < c \le 1 - \Omega(1)$, then there is a parameter $\Upsilon_{\#} = c^{-\binom{k}{s}}(\log n)^{O(1)}$ such that the following holds. If there is a randomized algorithm counting k-cliques in time $O(n^t)$ with error probability less than $1/\Upsilon_{\#}$ on hypergraphs drawn from $G(n, c, s)$, then there is a randomized algorithm counting k-cliques on worst-case s-uniform hypergraphs with error probability less than $1/3$ running in time $O\left(\Upsilon_{\#} \cdot n^{\max\{t,s\}}\right)$.*

We discuss the necessity of the error tolerance and the multiplicative slowdown in our worst-case to average-case reduction in section 2.2. This result has a number of

consequences for basing the average-case fine-grained complexity of k-clique counting over Erdős–Rényi hypergraphs on its worst-case complexity, which we now overview.

Counting k-cliques in worst-case hypergraphs is known to take $n^{\Omega(k)}$ time for randomized algorithms assuming the randomized Exponential Time Hypothesis (rETH)[2] if k does not grow with n [19, 18]. The best known worst-case algorithms up to subpolynomial factors are the $O\left(n^{\omega\lceil k/3\rceil}\right)$ time algorithm of [62] in the graph case of $s = 2$ and exhaustive $O(n^k)$ time search on worst-case hypergraphs with $s \geq 3$. Here, $\omega \leq 2.373$ denotes the matrix multiplication constant. Our reduction is the first worst-case to average-case reduction to Erdős–Rényi hypergraphs. It has different implications for the cases of dense and sparse hypergraphs, because of the factor $\Upsilon_{\#}$, as described next:

1. *Dense Erdős–Rényi graphs and hypergraphs.* When k and c are constant, our reduction constructs an efficient k-clique counting algorithm that succeeds on a worst-case input hypergraph with high probability, using $\text{polylog}(n)$ queries to an average-case oracle that correctly counts k-cliques on a $1-1/\text{polylog}(n)$ fraction of Erdős–Rényi hypergraphs drawn from $G(n, c, s)$. This essentially shows that k-clique counting in the worst case matches that on dense Erdős–Rényi hypergraphs. More precisely, k-clique counting on $G(n, c, s)$ with k, c, and s constant must take $\tilde{\Omega}\left(n^{\omega\lfloor k/3\rfloor}\right)$ 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 rETH-hard to count k-cliques in $n^{o(k)}$ time on $G(n, c, s)$ with k, c, and s constant.
2. *Sparse Erdős–Rényi graphs and 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 expectation of the quantity being counted, the number of k-cliques in $G(n, c, s)$, up to $\text{polylog}(n)$ factors.[3]

Precise statements of our results can be found in section 2.2. For simplicity, our results should be interpreted as applying to algorithms that succeed with probability $1 - (\log n)^{-\omega(1)}$ in the dense case and $1 - n^{-\omega(1)}$ in the sparse case.

We also give a second worst-case to average-case reduction for computing the parity of the number of k-cliques which has a weaker requirement of $1 - \Theta_{k,s}(1)$ on the error probability for the blackbox solving the problem on $G(n, c, s)$ in the dense case of $c = 1/2$. We provide an overview of our multistep worst-case to average-case reduction in section 1.2. The steps are described in detail in section 3.

Algorithms for k-clique counting on $G(n, c, s)$. We also analyze several natural algorithms for counting k-cliques in sparse Erdős–Rényi hypergraphs. These include an extension of the natural greedy algorithm mentioned previously from k-CLIQUE to counting k-cliques, a modification to this algorithm using the matrix multiplication step of [62] and an iterative algorithm achieving nearly identical guarantees. These algorithms count k-cliques in $G(n, c, s)$ when $c = \Theta(n^{-\alpha})$ with several different run-

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

[3] For the subclass of algorithms that enumerate k-cliques one by one, the k-clique count is a trivial lower bound on the runtime. Our general lower bound matches this heuristic lower bound.

times, the best of which are as follows:

- $\tilde{O}\big(n^{k+1-\alpha\binom{k}{s}}\big)$ if $s \geq 3$ and $k < \tau + 1$;
- $\tilde{O}\big(n^{\tau+2-\alpha\binom{\tau+1}{s}}\big)$ if $s \geq 3$ and $\tau + 1 \leq k \leq \kappa + 1$; and
- $\tilde{O}\big(n^{\omega\lceil k/3\rceil+\omega-\omega\alpha\binom{\lceil k/3\rceil}{2}}\big)$ 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 $\alpha\binom{\kappa}{s-1} < s$. The thresholds κ and τ have natural interpretations as roughly the clique number and most frequent clique size in the graph $G(n, c, s)$, respectively. Throughout, 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 $1/\mathrm{poly}(n)$.

The threshold $\tau + 1$ also has a natural interpretation as the k-clique percolation threshold [26, 63, 28], defined below. 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 $G(n, c)$, the authors of [26] 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 [13]. In the graph case of $s = 2$, this threshold matches $\tau + 1$, which is the largest integer k such that $\alpha < \frac{1}{k-1}$. Following the same heuristic as in [26], our threshold $\tau + 1$ is a natural extension of the k-clique percolation threshold to the hypergraph case of $s \geq 3$. In other words, $\tau + 1$ roughly corresponds to the largest value of k at which a local search algorithm can explore all the cliques in the hypergraph starting from any given clique.

Comparing our upper and lower bounds. A comparison of our algorithmic guarantees and average-case lower bounds based on the 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.

Two interesting problems left open by 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 other open problems and some extensions of our methods are discussed in section 6.

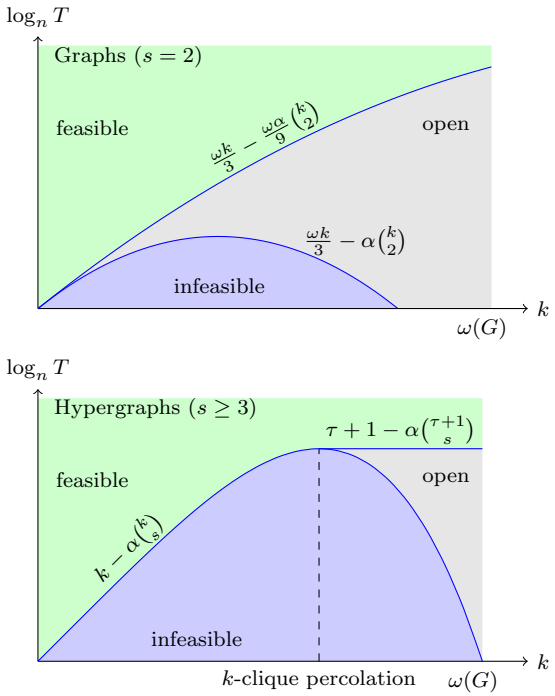

FIG. 1. *Comparison of our algorithms and average-case lower bounds for counting k-cliques in sparse Erdős–Rényi hypergraphs $G(n, c, s)$ with $c = \Theta(n^{-\alpha})$. Green denotes runtimes T feasible for each k, blue denotes T infeasible given that the best known worst-case algorithms are optimal,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

1.2. 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 [57, 35, 40, 41] 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.

Random self-reducibility for counting k-cliques. 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

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

This polynomial has been used in a number of papers, including by Goldreich and Rothblum [43] to construct a distribution on dense graphs for which counting k-cliques is provably hard on average. However, their techniques are primarily focused on the error probability requirement for the average-case blackbox. As a result, the distribution they obtain is far from Erdős–Rényi and 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 $A \in \mathbb{F}_p^{\binom{n}{2}}$, Goldreich and Rothblum [43] use several gadgets (including replacing vertices by independent sets and taking disjoint unions of graphs) in order to create a larger unweighted random graph A' whose k-clique count is equal to $k! \cdot P(A) \pmod p$ for appropriate p. However, any nontrivial gadget-based reduction seems to have little hope of arriving at something close to the Erdős–Rényi distribution, because gadgets inherently create nonuniform structure.

Reducing to k-partite graphs. We instead consider a different polynomial for graphs on nk vertices with $nk \times nk$ adjacency matrix A,

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

The polynomial P' correctly counts the number of k-cliques if A is k-partite with vertex k-partition $[n] \sqcup ([2n] \setminus [n]) \sqcup \cdots \sqcup ([kn] \setminus [(k-1)n])$. We first reduce clique counting in the worst case to computing P' in the worst case; this is a simple step, because it is a purely worst-case reduction. Next, we construct a recursive counting procedure that reduces evaluating P' on Erdős–Rényi graphs to counting k-cliques in Erdős–Rényi graphs. Therefore, it suffices to prove that if evaluating P' is hard in the worst case, then evaluating P' on Erdős–Rényi graphs is also hard.

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

Eliminating weights with random sparse binary expansions. We solve this by decomposing the random weighted graph $X \sim \mathrm{Unif}[\mathbb{F}_p^{\binom{nk}{2}}]$ into a weighted sum of graphs $Z^{(0)}, \ldots, Z^{(t)} \in \{0,1\}^{\binom{nk}{2}}$ such that each $Z^{(i)}$ is close to Erdős–Rényi $G(n,c,2)$. Specifically, this additive decomposition satisfies $X \equiv \sum_{i=0}^{t} 2^i Z^{(i)} \pmod p$, i.e., that we can write X as a binary expansion modulo p of Erdős–Rényi graphs. Importantly, in section 4 we derive near-optimal bounds on t and prove that we can take t to be quite small, growing only as $\mathrm{poly}(c^{-1}(1-c)^{-1}\log(p))$. This technique seems likely to have applications elsewhere. For the unbiased case of $c = 1/2$, a version of this binary expansion technique appeared previously in [44].

Now, using the binary expansion decomposition of X, we algebraically manipulate

P' as follows:

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

Here, $Z^{(f)}$ is the nk-vertex graph with entries given by $Z^{(f_{\bar{a}\bar{b}})}_{ab}$ for $1 \le a < b \le nk$, 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 $G(n, c, 2)$. This concludes our reduction.[4]

We remark that an important difference between our reduction and the reduction in [43] is the number of and structure of the calls to the average-case blackbox. Our reduction requires many successful calls to the blackbox in order to obtain a single correct evaluation of the polynomial $P'(A)$, which is where our low error probability requirement comes from. The gadgets in [43] are specifically designed to only require a single successful call to obtain a single correct evaluation of $P(A)$. Thus, even given a blackbox with a constant error probability, the Berkelamp–Welch algorithm can recover $P(A)$ in the case of [43].

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 2.2 and 3.

1.3. 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 $\mathsf{P}^{\#\mathsf{P}} = \mathsf{BPP}$ [73, 17]. (Under the slightly stronger assumption that $\mathsf{P}^{\#\mathsf{P}} \neq \mathsf{AM}$, and with different techniques, the authors of [33] proved that computing the permanent on large finite fields is hard even with exponentially small success probability.) Recently, the authors of [8] 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

[4] If we had instead worked with P, then this argument would fail. The argument uses the k-partiteness structure of P' as follows: for every pair of vertices $a, b \in [nk]$ and $f \in \{0,\ldots,t\}^{\binom{k}{2}}$, the term $Z^{(f_{ij})}_{ab}$ appearing in the sum is uniquely determined by $a \in [ik]\setminus[(i-1)k]$ and $b \in [jk]\setminus[(j-1)k]$. So given f we can define a graph $Z^{(f)}$ uniquely. On the other hand, running the same argument with the polynomial P, the term $Z^{(f_{ij})}_{ab}$ for many different i, j would appear in the sum, and there is no way to uniquely define a graph $Z^{(f)}$.

used by Gamarnik and Kızıldağ [37] in order to prove that exactly computing the partition function of the Sherrington–Kirkpatrick model in statistical physics is hard on average.

If a problem is random self-reducible, then random instances of the problem are essentially as hard as worst-case instances, and therefore one may generate a hard instance of the problem by simply generating a random instance. Because of this, random self-reducibility plays an important role in cryptography: it allows one to base cryptographic security on random instances of a problem, which can generally be generated efficiently. A prominent example of a random self-reducible problem with applications to cryptography is the problem of finding a short vector in a lattice. In a seminal paper, Ajtai [1] gave a worst-case to average-case reduction for this short-vector problem. His ideas were subsequently applied to prove the average-case hardness of the learning with errors (LWE) problem, which underlies lattice cryptography [1, 67]. A good survey covering worst-case to average-case reductions in lattice cryptography is [68].

There are known restrictions on problems that are self-reducible. For example, nonadaptive worst-case to average-case reductions for NP-complete problems fail unless $\mathsf{coNP} \subseteq \mathsf{NP/poly}$ [35, 12, 11].

Subsequent work. Several new results have been proved subsequent to the first appearance of our work. Goldreich [42] provided a simpler reduction for counting the parity of the number of cliques in the uniform $G(n, 1/2)$ Erdős–Rényi graph case. Goldreich obtained error tolerance $\exp(-k^2)$ in this case, which is an improvement over the error tolerance $\exp(-\tilde{O}(k^2))$ in our Theorem 2.9. Hirahara and Shimizu [47] studied the average-case complexity of counting bicliques in uniformly random bipartite graphs, obtaining near-optimal runtime bounds assuming the Strong Exponential Time Hypothesis (SETH). And Dalirrooyfard, Lincoln, and Vassilevska Williams [23] extended our techniques to obtain average-case hardness for counting the number of copies of any graph H as an induced subgraph of an Erdős–Rényi graph $G(n, 1/2)$; they also used these techniques to show that simple variations of the orthogonal vectors, 3-sum and zero-weight k-clique problems, are hard to count on average for uniform inputs.

1.4. Notation and preliminaries. An 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 runtime of an algorithm A on inputs of size parametrized by n; for simplicity, we assume throughout that $T(A, n)$ is nondecreasing in n. All algorithms in this paper are randomized, and each (possibly biased) coin flip incurs constant computational cost.

2. Problem formulations and average-case lower bounds.

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

DEFINITION 2.1. $\#(k,s)$-CLIQUE *denotes the problem of counting the number of k-cliques in an s-uniform hypergraph G.*

DEFINITION 2.2. PARITY-(k,s)-CLIQUE *denotes the problem of counting the number of k-cliques up to parity in an s-uniform hypergraph G.*

DEFINITION 2.3. 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 in the worst-case setting. When k is allowed to be an unbounded input to the problem, DECIDE-(k,s)-CLIQUE is known to be NP-complete [53] and $\#(k,s)$-CLIQUE is known to be #P-complete [74]. 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 worst-case reduction from PARITY-(k,s)-CLIQUE to $\#(k,s)$-CLIQUE is immediate. As we show in Appendix A, in the worst-case setting, 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 [76]. In the graph case of $s = 2$, the fastest known algorithms for all of these problems take $\Theta(n^{\omega\lceil k/3\rceil})$ time, where $2 \leq \omega < 2.4$ is the fast matrix multiplication constant [48, 62]. 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 [66] proves that monotone circuits require $\tilde{\Omega}(n^k)$ operations to solve DECIDE-$(k,2)$-CLIQUE in the case of constant k. Monotone circuit lower bounds are also known in the case when $k = k(n)$ grows with n [2, 4]. In [29], 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 $f(k) \cdot \text{poly}(n)$—then any algorithm in the parametrized complexity class W[1] is also fixed-parameter-tractable. This provides further evidence that DECIDE-$(k,2)$-CLIQUE is intractable for large k. Finally, the authors of [19] show that solving DECIDE-$(k,2)$-CLIQUE in $n^{o(k)}$ time is ETH-hard for constant k.[5] We therefore conjecture that the k-clique problems take $n^{\Omega(k)}$ time on worst-case inputs when k is constant, as formalized below.

CONJECTURE 2.4 (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 2.5 (worst-case hardness of PARITY-(k,s)-CLIQUE). *Let k be con-*

[5] 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 an 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.

stant. 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 2.6 (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 2.6 is implied by rETH, they all follow from rETH. We also formulate a stronger version of the clique-counting hardness conjecture, which asserts that the current best known algorithms for k-clique counting are optimal.

Conjecture 2.7 (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 $s\geq 3$.*

2.2. Average-case lower bounds for counting k-cliques in $G(n,c,s)$. Our first main result is a worst-case to average-case reduction solving either $\#(k,s)$-clique or Parity-(k,s)-clique on worst-case hypergraphs given a blackbox solving the problem on *most* Erdős–Rényi hypergraphs drawn from $G(n,c,s)$. We discuss this error tolerance over sampling Erdős–Rényi hypergraphs as well as the multiplicative overhead in our reduction below. These results show that solving the k-clique problems on Erdős–Rényi hypergraphs $G(n,c,s)$ is as hard as solving them on worst-case hypergraphs for certain choices of k,c, and s. Therefore, the worst-case hardness assumptions, Conjectures 2.4, 2.5, and 2.7, imply average-case hardness on Erdős–Rényi hypergraphs for $\#(k,s)$-clique and Parity-(k,s)-clique.

Theorem 2.8 (worst-case to average-case reduction for $\#(k,s)$-clique). *There is an absolute constant $C>0$ such that if we define*

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

*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*

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

where $T(\mathcal{A},\ell)$ denotes the runtime of algorithm $\mathcal{A}$ on ℓ-vertex hypergraphs.

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

Theorem 2.9 (worst-case to average-case reduction for Parity-(k,s)-clique). *We have the following:*

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

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

then the following statement holds. Let A be a randomized algorithm for Parity-(k,s)-clique *with error probability less than $1/\Upsilon_{P,1}$ on hypergraphs*

drawn from $G(n,c,s)$. Then there exists an algorithm B for Parity-(k,s)-clique *that has error probability less than $1/3$ on any hypergraph, such that*

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

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*

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

Our worst-case to average-case reductions yield the following fine-grained average-case lower bounds for k-clique counting and parity on Erdős–Rényi hypergraphs based on Conjectures 2.4 and 2.7. 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 2.10 (average-case hardness of $\#(k,s)$-clique on dense $G(n,c,s)$). *If $k,c,\epsilon > 0$ are constant, then we have the following:*

1. *Assuming Conjecture* 2.4*, 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 $T(A,n) \ge n^{\Omega(k)}$.*
2. *Assuming Conjecture* 2.7*, 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 $T(A,n) \ge \tilde{\Omega}\left(n^{\omega\lceil k/3\rceil}\right)$ if $s = 2$ and $T(A,n) \ge \tilde{\Omega}(n^k)$ if $s \ge 3$.*

Corollary 2.11 (average-case hardness of $\#(k,s)$-clique on sparse $G(n,c,s)$). *Let $k,\alpha,\epsilon > 0$ be constants, and let $c = \Theta(n^{-\alpha})$. Assuming Conjecture* 2.7*, 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 $G(n,c,s)$ must have runtime at least $T(A,n) \ge \tilde{\Omega}\big(n^{\omega\lceil k/3\rceil-\alpha\binom{k}{s}}\big)$ if $s = 2$ and $T(A,n) \ge \tilde{\Omega}\big(n^{k-\alpha\binom{k}{s}}\big)$ if $s \ge 3$.*

We remark that Conjecture 2.4 implies there is a constant $C > 0$ such that a version of Corollary 2.11 holds with the weaker conclusion that $T(A,n) \ge n^{\Omega(k)}$ for any $\alpha \le Ck/\binom{k}{s}$. For Parity-$(k,s)$-clique, we consider here the implications of Theorem 2.9 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 2.12 (average-case hardness of Parity-(k,s)-clique on $G(n,1/2,s)$). *Let k be constant. Assuming Conjecture* 2.5*, 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 $T(A,n) \ge n^{\Omega(k)}$.*

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

$\alpha \le k\binom{k}{s}^{-1} = s\binom{k}{s-1}^{-1}$, which is necessary for the expected number of k-cliques in $G(n,c,s)$ to not tend to zero. However, even when this expectation is decaying, the problem $\#(k,s)$-CLIQUE as we formulate it is still nontrivial. The simple algorithm that always outputs zero fails with a polynomially small probability that does not appear to meet the $1/\Upsilon_{\#}$ requirement in our worst-case to average-case reduction. A simple analysis of this error probability can be found in Lemma 5.1. Note that even when $\alpha > s\binom{k}{s-1}^{-1}$, GREEDY-RANDOM-SAMPLING and its derivative algorithms in section 5 still have guarantees and succeed 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 5 imply lower bounds on the multiplicative overhead factor $\Upsilon_{\#}$ in Theorem 2.8. In the hypergraph case of $s \ge 3$ and below the k-clique percolation threshold, it must follow that the overhead is at least $\Upsilon_{\#} = \tilde{\Omega}\big(n^{\alpha\binom{k}{s}}\big) = \tilde{\Omega}\big(c^{-\binom{k}{s}}\big)$. Otherwise, our algorithms combined with our worst-case to average-case reduction would contradict Conjecture 2.7. Up to polylog(n) factors, this exactly matches the $\Upsilon_{\#}$ from our reduction. In the graph case of $s = 2$, it similarly must follow that the overhead is at least $\Upsilon_{\#} = \tilde{\Omega}\big(n^{\frac{\omega\alpha}{9}\binom{k}{s}}\big) = \tilde{\Omega}\big(c^{-\frac{\omega}{9}\binom{k}{s}}\big)$ to not contradict Conjecture 2.7. 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 2.8 and 2.9 require that the error of the average-case blackbox on Erdős–Rényi hypergraphs go to zero as k goes to infinity. This error tolerance requirement is unavoidable. When $k = \omega(\log n)$ in the dense Erdős–Rényi graph case of $G(n,1/2)$, there is a k-clique with at most $\binom{n}{k}2^{-\binom{k}{2}} = o(1)$ probability by a union bound on k-subsets of vertices. So in this regime clique counting on $G(n,1/2)$ with constant error probability is not hard: the algorithm that always outputs zero achieves $o(1)$ average-case error.

If $k \triangleq 3\log_2 n$, then the probability of a k-clique on $G(n,1/2)$ is less than $\binom{n}{k}2^{-\binom{k}{2}} \le 2^{-k^2/6}$. So average-case k-clique counting is not hard with error more than $2^{-k^2/6}$. On the other hand, our $\#(k,2)$-CLIQUE reduction works with average-case error less than $1/\Upsilon_{\#} = 2^{-\Omega(k^2 \log\log n)}$. And our PARITY-$(k,2)$-CLIQUE reduction is more lenient, requiring error only less than $2^{-\Omega(k^2 \log\log\log n)}$. 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. The probability that a $G(n,c)$ graph contains a k-clique is less than $(nc^{(k-1)/2})^k$. 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 scales as $n^{-O(k^2 \log\log n)}$.

3. Worst-case to average-case reduction for $G(n, c, s)$. In this section, we give our main worst-case to average-case reduction that transforms a blackbox solving $\#(k, s)$-CLIQUE on $G(n, c, s)$ into a blackbox solving $\#(k, s)$-CLIQUE on a worst-case input hypergraph. This also yields a worst-case to average-case reduction for PARITY-(k, s)-CLIQUE and proves Theorems 2.8 and 2.9. The reduction involves the following five main steps, the details of which are in sections 3.1 to 3.5.

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

These steps are combined in section 3.6 to complete the proofs of Theorems 2.8 and 2.9. 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 3.1 (k-partite hypergraphs). *Given an 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 $e = \{u_1, \ldots, u_s\} \in E(G)$ the labels $L(u_1), L(u_2), \ldots, L(u_s)$ 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

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

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 the size of this set of permitted hyperedges. Thus, we will refer to elements $x \in \{0, 1\}^N$ and k-partite s-uniform hypergraphs on nk vertices interchangeably. This definition also extends to Erdős–Rényi hypergraphs.

DEFINITION 3.2 (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 each hyperedge $e = \{u_1, \ldots, u_s\} \in E(G)$ with probability c for all e with $L(u_1), L(u_2), \ldots, L(u_s)$ distinct.*

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

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

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

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

   For each $1 \le t \le T$, compute $P_{n,k,s}(G') \pmod{p_t}$, as follows:
   (1) Use the procedure of [41] in order to reduce the computation of $P_{n,k,s}(G') \pmod{p_t}$ to the computation of $P_{n,k,s}$ on $M = 12\binom{k}{s}$ distinct inputs $x_1,\dots,x_M \sim \mathrm{Unif}[\mathbb{F}_{p_t}^N]$.
   (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 3.8 in order to sample $(\tilde{Z}^{(0)},\dots,\tilde{Z}^{(B)})$ close to $(\mathrm{Ber}(c)^{\otimes N})^{\otimes B}$ in total variation distance, such that $x_m \equiv \sum_{b=0}^{B} 2^b \cdot \tilde{Z}^{(b)} \pmod{p_t}$. It suffices to take $B = \Theta(c^{-1}(1-c)^{-1} s(\log n)(\log p_t))$.
      (ii) For each function $a : \binom{[k]}{s} \to \{0,\dots,B\}$, define $\tilde{Z}_S^{(a \circ L)} = \tilde{Z}^{a(L(S))}$ for all $S \in [N] \subset \binom{[n]}{s}$. Note that for each a, the corresponding $\tilde{Z}^{(a \circ L)}$ is approximately distributed as $\mathrm{Ber}(c)^{\otimes N}$. Use algorithm A and the recursive counting procedure of Lemma 3.10 in order to compute $P_{n,k,s}(\tilde{Z}^{(a \circ L)})$ for each a.
      (iii) Set $P_{n,k,s}(G') \leftarrow \sum_{a : \binom{[k]}{s} \to \{0,\dots,B\}} 2^{|a|_1} \cdot P_{n,k,s}(\tilde{Z}^{(a \circ L)})$.
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 $P_{n,k,s}(G')$.

FIG. 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.

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 $\mathrm{Ber}(c)^{\otimes N}$.

3.1. Worst-case reduction to k-partite hypergraphs. In the next lemma, we prove that the worst-case complexity of $\#(k,s)$-CLIQUE and PARITY-(k,s)-CLIQUE is nearly unaffected when we restrict the inputs to be worst-case k-partite hypergraphs. This step is important, because the special structure of k-partite hypergraphs

will simplify future steps in our reduction.

LEMMA 3.3. *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 exists an algorithm B for $\#(k,s)$-CLIQUE with error probability less than $1/3$ on any hypergraph G satisfying a runtime upper bound $T(B,n) \le T(A,n) + O(k^s n^s)$. 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

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

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

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

where $v_1 < v_2 < \cdots < v_k$. 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 3.3 is that any worst-case hardness for $\#(k,s)$-CLIQUE and PARITY-(k,s)-CLIQUE on general s-uniform hypergraphs immediately transfers to the k-partite case. For instance, the lower bounds of Conjectures 2.4, 2.5, and 2.7 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.

3.2. Counting k-cliques as a low-degree polynomial. A key step in our worst-case to average-case reduction is to express the number of k-cliques as a low-degree polynomial in the adjacency matrix. As mentioned in the introduction, a similar step—but without the k-partiteness constraint—appears in the worst-case to average-case reduction of Goldreich and Rothblum [43].

Let $\mathcal{E} \subset \binom{V(G)}{s}$ be the set of possible hyperedges that respect the k-partition: i.e., $\mathcal{E} = \{A \in \binom{V(G)}{s} : |L(A)| = s\}$. Let $N \triangleq N(n,k,s) = |\mathcal{E}|$, and identify $\mathcal{E}$ with $[N]$ through a bijection $\pi : [N] \to \mathcal{E}$. To simplify the notation, we will omit the map π in the proof and simply treat $[N]$ and $\mathcal{E}$ as the same set. Thus, each $x \in \{0,1\}^N$ corresponds to a k-partite hypergraph where x_A is the indicator that $A \in \mathcal{E}$ is an edge in the hypergraph. The number of k-cliques of a k-partite hypergraph $x \in \{0,1\}^N$ is a degree-D polynomial $P_{n,k,s} : \{0,1\}^N \to \mathbb{Z}$, where $D \triangleq D(k,s) = \binom{k}{s}$:

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

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

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

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 3.4. *Let $x \in \{0,1\}^N$ denote an s-uniform hypergraph that is k-partite with vertex labelling L. Let $p_1, p_2, \ldots, p_t$ be t distinct primes, such that $\prod_i p_i \in (n^k, n^{2k})$. First, solving $\#(k,s)$-CLIQUE reduces to computing $P_{n,k,s,\mathbb{F}_{p_i}}(x)$ for all $i \in [t]$, plus $O((k \log n)^2)$ additive computational overhead. Second, 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. For any $i \in [t]$, it holds that $P_{n,k,s,\mathbb{F}_{p_i}}(x) \equiv P_{n,k,s}(x) \pmod{p_i}$, which proves the second item of the proposition. The first item follows since $P_{n,k,s}(x) \le n^k$, because there are at most n^k cliques in the hypergraph. Thus, $P_{n,k,s}(x)$ can be reconstructed from $P_{n,k,s}(x) \pmod{p_i}$ for all $i \in [t]$ in time $O((k \log n)^2)$ by the computational version of the Chinese remainder theorem (Theorem 4.6 of [72]). ∎

PROPOSITION 3.5. *Let $\mathbb{F}$ be a finite field of characteristic 2. Let $x \in \{0,1\}^N$ be an 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 $P_{n,k,s,\mathbb{F}}(x) \equiv P_{n,k,s}(x) \pmod{\mathrm{char}(\mathbb{F})}$. ∎

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

The following well-known lemma states the random self-reducibility of low-degree polynomials. The lemma first appeared in [41]. We follow the proof of [8] in order to present the lemma with explicit guarantees on the running time of the reduction.

LEMMA 3.6 (Theorem 4 of [41]). *Let $\mathbb{F}$ be a finite field with $|\mathbb{F}| = q$ elements. Let $N > 0$ and $1 \le D < q/12$. 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 $O((N+D^2)D \log^2 q + T(A,N) \cdot D)$ 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 3.6 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 $\mathbb{F}^N$.

3.4. Reduction to evaluating the polynomial on $G(nk, c, s, k)$. So far, we have reduced worst-case clique counting over unweighted hypergraphs to the average-case problem of computing $P_{n,k,s,\mathbb{F}}$ over k-partite hypergraphs with random edge weights in $\mathbb{F}$. It remains to reduce from computing $P_{n,k,s,\mathbb{F}}$ on inputs $x \sim \mathrm{Unif}\left[\mathbb{F}^N\right]$ 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 expansions modulo p to Unif$[\mathbb{F}_p]$ and related algorithmic corollaries from section 4.

We first present the reduction that will be applied in the case of PARITY-(k,s)-CLIQUE. Recall $D = \binom{k}{s}$ is the degree of $P_{n,k,s}$. The following lemma will be used only for the PARITY-(k,s)-CLIQUE case.

LEMMA 3.7. *Let p be prime, and let $t \geq 1$. Suppose A is an algorithm that computes $P_{n,k,s,\mathbb{F}_p}(y)$ with error probability less than $\delta \triangleq \delta(n)$ for $y \sim \text{Unif}\left[\mathbb{F}_p^N\right]$ in time $T(A,n)$. Then there exists an algorithm B that computes $P_{n,k,s,\mathbb{F}_{p^t}}(x)$ with error probability less than $t^D \cdot \delta$ for $x \sim \text{Unif}\left[\mathbb{F}_{p^t}^N\right]$ in time $T(B,n) = O\big(Nt^4(\log p)^3 + t^D \cdot T(A,n)\big)$.*

Proof. We give a reduction computing $P_{n,k,s,\mathbb{F}_{p^t}}(x)$ where $x \sim \text{Unif}\left[\mathbb{F}_{p^t}^N\right]$ 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 is able to locate 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, Driscoll, and Shallit [7]. 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 $\mathbb{F}_p$.[6] 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 i.i.d. according to Unif $\left[\mathbb{F}_p^N\right]$.

Given a coloring of the hyperedges $b : [N] \to \{0, 1, \ldots, t-1\}$, 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 coloring b, the vector $x^{(b)}$ is uniform in $\mathbb{F}_p^N$.

In our proof, for every map $a : \binom{[k]}{s} \to \{0, 1, \ldots, t-1\}$, we construct a coloring $a \circ L : [N] \to \{0, \ldots, t-1\}$ of the hyperedges $[N]$ using the k-partiteness of the hypergraph. Given a hyperedge $W = \{w_1, \ldots, w_s\} \in \mathcal{E} = [N]$, we have that $L(W) \in \binom{[k]}{s}$ by the k-partiteness of the hypergraph, and hence the color $(a \circ L)(W) \triangleq a(L(W))$ is well-defined. As above, for any fixed a, the vector $x^{(a \circ L)}$ is uniform in $\mathbb{F}_p^N$.

We now manipulate $P_{n,k,s,\mathbb{F}_{p^t}}$. First we write each entry x_{u_S} in the normal basis, and then we redistribute terms to write $P_{n,k,s,\mathbb{F}_{p^t}}$ as a weighted sum of clique counts modulo p:

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

[6] For a good survey on normal bases, we recommend [39].

$$= \sum_{a:\binom{[k]}{s}\to\{0,\dots,t-1\}} \left(\prod_{S\in\binom{[k]}{s}} \beta^{p^{a(S)}}\right)\left(\sum_{\substack{\{u_1,\dots,u_k\}\subset V(G)\\ \forall i\ L(u_i)=i}} \prod_{S\in\binom{[k]}{s}} x_{u_S}^{(a(S))}\right)$$

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

Since $x^{(a\circ L)} \sim \mathrm{Unif}\left[\mathbb{F}_p^N\right]$ for each fixed map a, computing $P_{n,k,s,\mathbb{F}_{p^t}}(x)$ reduces to evaluating $P_{n,k,s,\mathbb{F}_p}$ on t^D uniformly random inputs in $\mathbb{F}_p^N$ and outputting a weighted sum of the evaluations. The error probability is bounded by 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. One of the main lemmas driving the reduction is the following.

LEMMA 3.8. *There exists an absolute constant $K>0$ such that the following holds. Let $p>2$ be prime, let $\epsilon>0$, let $c\in(0,1)$, and let $t \geq K\cdot c^{-1}(1-c)^{-1}\log(p/\epsilon)\log p$. Then there exists an $O(pt\log(1/\epsilon)\log(p))$-time algorithm that, given $x\in\mathbb{F}_p$, samples a random variable $\tilde{Z}_x = (\tilde{Z}_x^{(0)},\dots,\tilde{Z}_x^{(t-1)}) \in \{0,1\}^t$ satisfying $\sum_{i=0}^{t-1} 2^i\cdot\tilde{Z}_x^{(i)} \equiv x \pmod p$ almost surely. Moreover, if $x\sim\mathrm{Unif}[\mathbb{F}_p]$, then $d_{TV}(\mathcal{L}(\tilde{Z}_x),\mathrm{Ber}(c)^{\otimes t})\leq\epsilon$.*

The proof of Lemma 3.8 is deferred to section 4. It is a central ingredient in the $\#(k,s)$-CLIQUE reduction and will be used through the following lemma.

LEMMA 3.9. *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 $x\sim\mathrm{Unif}\left[\mathbb{F}_p^N\right]$ in time upper bounded by $T(B,n)=O\left(Npt\log(Np/\gamma)\log(p)+t^D\cdot T(A,n)\right)$.*

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 3.8 to sample $\tilde{Z}_j=(\tilde{Z}_j^{(0)},\tilde{Z}_j^{(1)},\dots,\tilde{Z}_j^{(t-1)})\in\{0,1\}^t$ satisfying

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

By Lemma 3.8, we may choose $t=O(c^{-1}(1-c)^{-1}\log(Np/\gamma)\log p)$, and this sampling can be carried out in $O(Npt\log(Np/\gamma)\log(p))$ time. Now expand $P_{n,k,s,\mathbb{F}_p}(x)$ in terms of $\tilde{Z}$, similarly to the calculations in Lemma 3.7. We are working in $\mathbb{F}_p$, so the following equalities hold modulo p:

$$P_{n,k,s,\mathbb{F}_p}(x) = \sum_{\substack{\{u_1,\dots,u_k\}\subset V(G)\\ \forall j\ L(u_j)=j}} \prod_{S\in\binom{[k]}{s}} x_{u_S}$$

$$= \sum_{\substack{\{u_1,\dots,u_k\}\subset V(G)\\ \forall j\ L(u_j)=j}} \prod_{S\in\binom{[k]}{s}} \left(\sum_{i=0}^{t-1} 2^i\cdot\tilde{Z}_{u_S}^{(i)}\right)$$

$$= \sum_{a:\binom{[k]}{s}\to\{0,\dots,t-1\}} \left(\sum_{\substack{\{u_1,\dots,u_k\}\subset V(G)\\ \forall i\ L(u_i)=i}} \prod_{S\in\binom{[k]}{s}} \left(2^{a(S)}\cdot \tilde{Z}^{(a(S))}_{u_S}\right)\right)$$

$$= \sum_{a:\binom{[k]}{s}\to\{0,\dots,t-1\}} \left(\prod_{S\in\binom{[k]}{s}} 2^{a(S)}\right)\left(\sum_{\substack{\{u_1,\dots,u_k\}\subset V(G)\\ \forall i\ L(u_i)=i}} \prod_{S\in\binom{[k]}{s}} \tilde{Z}^{(a(S))}_{u_S}\right)$$

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

where, as in the proof of Lemma 3.7, given any coloring $b : [N] \to \{0,\dots,t-1\}$, we define $\tilde{Z}^{(b)} \in \{0,1\}^N$ by $\tilde{Z}^{(b)}_j = \tilde{Z}^{(b(j))}_j$ for all $j \in [N]$. Computing $P_{n,k,s,\mathbb{F}_p}(x)$ thus reduces to computing a weighted sum over the t^D evaluations of $P_{n,k,s,\mathbb{F}_p}(\tilde{Z}^{(a\circ L)})$ for all maps $a : \binom{[k]}{s} \to \{0,\dots,t-1\}$. Our algorithm uses the blackbox A to compute each term and outputs the weighted sum. In other words, our algorithm returns

$$\sum_{a:\binom{[k]}{s}\to\{0,\dots,t-1\}} \left(\prod_{S\in\binom{[k]}{s}} 2^{a(S)}\right) A(\tilde{Z}^{(a\circ L)}).$$

Let E be the event that the calls to the blackbox are all correct; i.e., $A(\tilde{Z}^{(a\circ L)}) = P_{n,k,s,\mathbb{F}_p}(\tilde{Z}^{(a\circ L)})$ for all $a : \binom{[k]}{s} \to \{0,\dots,t-1\}$. If E holds, then our algorithm correctly computes $P_{n,k,s,\mathbb{F}_p}(x)$. It suffices to prove that

$$\mathbb{P}[E] > 1 - \gamma + t^D \cdot \delta.$$

For the analysis, note that for each $j \in [N]$, the random vector $(\tilde{Z}^{(0)}_j,\dots,\tilde{Z}^{(t-1)}_j)$ may be coupled with $(Z^{(0)}_j,\dots,Z^{(t-1)}_j) \sim \mathrm{Ber}(c)^{\otimes t}$, such that

$$\mathbb{P}[\tilde{Z}^{(i)}_j = Z^{(i)}_j\ \forall i,j] \geq 1-\gamma.$$

Moreover, since $\tilde{Z}^{(i)}_j$ is independent of $\tilde{Z}^{(k)}_l$ whenever $j \neq l$, in the coupling we may choose Z such that $Z^{(i)}_j$ is independent of $Z^{(k)}_l$ whenever $j \neq l$. Thus, for any fixed coloring $b : [N] \to \{0,\dots,t-1\}$, the entries $Z^{(b)}_1,\dots,Z^{(b)}_N$ are independent and distributed as $\mathrm{Ber}(c)$. In other words, $Z^{(b)} \sim G(nk,c,s,k)$. We use these facts to lower bound the probability of E as follows:

$$\begin{aligned}
\mathbb{P}[E] &\geq \mathbb{P}[E \text{ and } \tilde{Z}^{(i)}_j = Z^{(i)}_j\ \forall i,j]\\
&= \mathbb{P}[A(\tilde{Z}^{(a\circ L)}) = P_{n,k,s,\mathbb{F}_p}(\tilde{Z}^{(a\circ L)})\ \forall a,\ \text{and } \tilde{Z}^{(i)}_j = Z^{(i)}_j\ \forall i,j]\\
&= \mathbb{P}[A(Z^{(a\circ L)}) = P_{n,k,s,\mathbb{F}_p}(Z^{(a\circ L)})\ \forall a,\ \text{and } \tilde{Z}^{(i)}_j = Z^{(i)}_j\ \forall i,j]\\
&\geq 1 - (1 - \mathbb{P}[\tilde{Z}^{(i)}_j = Z^{(i)}_j\ \forall i,j]) - \sum_{a:\binom{[k]}{s}\to\{0,\dots,t-1\}} \mathbb{P}[A(Z^{(a\circ L)}) \neq P_{n,k,s,\mathbb{F}_p}(Z^{(a\circ L)})]\\
&> 1 - \gamma - t^D\cdot\delta,
\end{aligned}$$

where the second-to-last line is a union bound, and the last line uses that $Z^{(a\circ L)} \sim G(nk, c, s, k)$ for any fixed a, and applies the error guarantee of A. This proves correctness of the algorithm 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 3.8. In this case, for each $j \in [N]$ we sample $\tilde{Z}_j = (\tilde{Z}_j^{(0)}, \dots, \tilde{Z}_j^{(t-1)}) \in \{0,1\}^t$ such that

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

By Lemma 4.4 (deferred but analogous to Lemma 3.8), we may choose $t = O(c^{-1}(1-c)^{-1}\log(N/\gamma))$, and we may sample in time $O(Nt\log(N/\gamma))$. By a similar, and simpler, calculation to the one for the case $p > 2$, we have that

$$P_{n,k,s,\mathbb{F}_2}(x) = \sum_{a:\binom{[k]}{s}\to\{0,\dots,t-1\}} P_{n,k,s,\mathbb{F}_2}(\tilde{Z}^{(a\circ L)}).$$

Our algorithm returns

$$\sum_{a:\binom{[k]}{s}\to\{0,\dots,t-1\}} A(\tilde{Z}^{(a\circ L)}),$$

which is correct with probability at least $1-\gamma-t^D\cdot\delta$ similarly to the $p > 2$ case. The proof is again to couple $\tilde{Z}$ with a random variable Z such that $\mathbb{P}[\tilde{Z}_j^{(i)} = Z_j^{(i)}\ \forall i, j] \ge 1-\gamma$, and, for each a, $Z^{(a\circ L)}$ is distributed as $G(nk, c, s, k)$. ∎

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

Lemma 3.10. *Let $\delta = \delta(n) \in (0,1)$ be a nonincreasing 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 $T(B, n) = O\left(2^k \cdot T(A, nk) + k^s n^s + s^2 k^3 2^k \log^2(nk)\right)$ 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 starting with G and adding every edge $e = \{v_1, v_2, \dots, v_s\} \in \binom{[n]\times[k]}{s}$ satisfying the condition $|\{L(v_1), \dots, L(v_s)\}| < s$ 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 $S \ne \emptyset$, it holds that $n \le |L^{-1}(S)| \le nk$ 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 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 $t_0, t_1, \ldots, t_d$, the next count t_{d+1} can be expressed by inclusion-exclusion as

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

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. The sizes of the integers manipulated are always at most $2^k \binom{nk}{s}$, so each arithmetic operation takes $O((ks \log(nk))^2)$ time. ∎

Repeating the same proof over $\mathbb{F}_2$ yields an analogue of Lemma 3.10 for Parity-(k, s)-clique, as stated below.

Lemma 3.11. *Lemma* 3.10 *holds when* $\#(k, s)$-clique *is replaced by* Parity-(k, s)-clique.

3.6. Proofs of Theorems 2.8 and 2.9. We now combine steps 1–5 formally in order to prove Theorems 2.8 and 2.9.

Proof of Theorem 2.8. Our goal is to construct an algorithm B solving $\#(k, s)$-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 $12\binom{k}{s} < p < 10 \log n^k$ computing $P_{n,k,s,\mathbb{F}_p}(y)$ 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 3.3 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 3.4 to construct A_0 from A_1 such that A_0 runs in time

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

The algorithm A_0 starts by using a sieve to find the first T primes $12\binom{k}{s} < p_1 < \cdots < p_T$ such that $\prod_{i=1}^{T} p_i > n^k$. Notice that $p_T \leq 10 \log n^k$, so this step takes time $O((\log n^k)^2)$. Then, given a k-partite hypergraph $x \in \{0,1\}^N$, algorithm A_0 computes $P_{n,k,s}(x)$ by first computing $P_{n,k,s,\mathbb{F}_{p_i}}(x)$ for all p_i with algorithm A_1, 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 $O(\log n^k)$. Finally, $P_{n,k,s}(x)$, the number of k-cliques in x, is computed from the values of $P_{n,k,s,\mathbb{F}_{p_i}}(x)$ in $O((k \log n)^2)$ time by the computational Chinese remainder theorem stated in Proposition 3.4.

3. *Reduce to computing $P_{n,k,s,\mathbb{F}_p}$ on random inputs in $\mathbb{F}_p^N$.* We use Lemma 3.6 to construct A_1 from A_2 such that A_1 runs in time

$$\begin{aligned} T(A_1, n) &= O((N + D^2) D \log^2 p + D \cdot T(A_2, n)) \\ &= O\left(n^s \binom{k}{s}^3 \log^2 \log n^k + \binom{k}{s} \cdot T(A_2, n) \right). \end{aligned}$$

4. *Reduce to computing $P_{n,k,s}$ on random inputs in $\{0,1\}^N$.* We use Lemma 3.9 to construct A_2 from A_3 such that A_2 runs in time

$$T(A_2, n) = O(N p t_p \log(Np) \log(p) + t_p^{\binom{k}{s}} \cdot T(A_3, n))$$

for some $t_p = O(c^{-1}(1-c)^{-1} s (\log n)(\log p))$. 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 $1/(4t_p^D) = 1/(4t_p^{\binom{k}{s}})$. Recall that we always have $p = O(k \log n)$ in this step, and hence t_p is upper bounded by a uniform value $t = \Theta(c^{-1}(1-c)^{-1} s (\log n)(\log k + \log \log n))$.

5. *Reduce to computing $\#(k,s)$-*CLIQUE *for $G(n,c,s)$ hypergraphs.* We use Lemma 3.10 to construct A_3 from A such that A_3 runs in time

$$T(A_3, n) = O((nk)^s + s^2 k^3 2^k \log^2(nk) + 2^k \cdot T(A, nk))$$

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 $\Upsilon_{\#} \geq (10t)^{\binom{k}{s}}$. In this case, since $\binom{k}{s} \geq k \geq 3$ without loss of generality, the error δ of A_3 will be at most $\delta \leq 2^k/\Upsilon_{\#} \leq 1/(5t)^{\binom{k}{s}} < 1/(4t^{\binom{k}{s}})$, which is what

we needed for the fourth step. It remains to put the runtime bounds together,

$$\begin{aligned}T(B,n) &= O\bigg((nk)^s + (\log n^k)^2 + (\log n^k)\cdot\bigg(n^s k^2\binom{k}{s}^3(\log n)^2 \\ &\qquad + \binom{k}{s}\cdot\Big(N(k\log n)t\log(N)\log(k\log n) + 4^k t^{\binom{k}{s}}\cdot(T(A,nk)+(nk)^s)\Big)\bigg)\bigg) \\ &= O\bigg(n^s k^3\binom{k}{s}^3(c^{-1}(1-c)^{-1})s(\log n)^4(\log k+\log\log n)^2 \\ &\qquad + (\log n)\cdot\Upsilon_{\#}\cdot(T(A,nk)+(nk)^s)\bigg) \\ &= O((\log n)\cdot(10t)^{\binom{k}{s}}\cdot n^s + (\log n)\cdot\Upsilon_{\#}\cdot(T(A,nk)+(nk)^s)),\end{aligned}$$

where we have used that $\binom{k}{s}\geq 3$ without loss of generality. The last term dominates since $\Upsilon_{\#}\geq(10t)^{\binom{k}{s}}$, and thus

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

This completes the proof. ∎

Proof of Theorem 2.9. The proof of item 1 of Theorem 2.9 is analogous to the proof of Theorem 2.8, except that it does not use the Chinese remainder theorem (Proposition 3.4). 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 3.3 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 3.5, if we can compute $P_{n,k,s,\mathbb{F}_{2^\kappa}}$ for worst-case inputs, then we can solve PARITY-(k,s)-CLIQUE. We use Lemma 3.6 to construct A_0 from A_1 such that A_0 runs in time

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

3. *Reduce to computing $P_{n,k,s,\mathbb{F}_2}$ on random inputs in $\mathbb{F}_2^N$.* We use Lemma 3.7 to construct A_1 from A_2 such that A_1 runs in time

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

and has error probability at most $\delta_2 \cdot \kappa^{\binom{k}{s}}$ on random inputs $w \sim \mathrm{Unif}[\mathbb{F}_{2^\kappa}^N]$. 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 3.9 to construct A_2 from A_3 such that A_2 runs in time

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

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

5. *Reduce to computing* Parity-(k,s)-clique *for $G(n,c,s)$ hypergraphs.* We use Lemma 3.11 to construct A_3 from A such that A_3 runs in time

$$T(A_3, n) = O((nk)^s + s^2k^3 2^k \log^2(nk) + 2^k \cdot T(A, nk))$$

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

As in the theorem statement, let

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

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,

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

if we choose $C > 0$ large enough. Since $\binom{k}{s} \ge k \ge 3$ without loss of generality, the second term dominates and

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

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 $T(B,n) = O(\Upsilon_{P,2}\cdot(T(A,nk) + (nk)^s))$. ∎

4. Random binary expansions modulo p. We fix some notation to be used throughout this section. Let p be a prime number, let $c \in (0, 1/2]$, and let $q_0, \dots, q_t \in [c, 1-c]$ be probabilities. Let $Z = (Z^{(0)}, \dots, Z^{(t)}) \in \{0,1\}^{t+1}$ be a vector of independent, biased Bernoulli random variables such that $Z^{(i)} \sim \mathrm{Ber}(q_i)$ for all $i \in \{0, \dots, t\}$. In this section, we consider the distributions of random binary expansions modulo p, of the form

$$Z^{(t)} \cdot 2^t + Z^{(t-1)} \cdot 2^{t-1} + \cdots + Z^{(0)} \pmod p.$$

We show that for t polylogarithmic in p, these distributions become close to uniformly distributed over $\mathbb{F}_p$. This is then used to go in the other direction, producing approximately independent Bernoulli variables that are the binary expansion of a number with a given residue. The special case of this argument in which the Bernoulli variables are unbiased has already appeared in an earlier work by Goldreich and Rothblum [44]. In that case, the proof of correctness is much simpler, because the Fourier-analytic tools used below can be avoided.

For $p > 2$, the main result of the section is the following slightly more general restatement of Lemma 3.8. It implies that we can efficiently sample biased binary expansions, conditioned on the expansion being equivalent to some x modulo p.

LEMMA 4.1 (restatement of Lemma 3.8). *There exists an absolute constant $K > 0$ such that the following holds. Let $p > 2$ be prime, let $\epsilon > 0$, and let $t \geq K \cdot c^{-1}(1-c)^{-1} \log(p/\epsilon) \log p$. Then there exists an $O(pt \log(1/\epsilon) \log(p))$-time randomized algorithm that, given $x \in \mathbb{F}_p$, outputs $\tilde{Z}_x = (\tilde{Z}_x^{(0)}, \dots, \tilde{Z}_x^{(t)}) \in \{0,1\}^{t+1}$ satisfying $\sum_{i=0}^t 2^i \cdot \tilde{Z}_x^{(i)} \equiv x \pmod p$ almost surely. Moreover, if $R \sim \mathrm{Unif}[\mathbb{F}_p]$, then $d_{TV}(\mathcal{L}(\tilde{Z}_R), \mathcal{L}(Z)) < \epsilon$.*

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 a simple claim showing that sufficiently long geometric progressions with ratio 2 in $\mathbb{F}_p$ contain a middle residue modulo p.

CLAIM 4.2. *Suppose that $a_1, \dots, a_k \in \mathbb{F}_p$ is a sequence with $a_1 \neq 0$ and $a_{i+1} = 2a_i$ for each $1 \leq i \leq k-1$. Then, if $k \geq 1 + \log_2(p/3)$, there is some j with $\frac{p}{3} \leq a_j \leq \frac{2p}{3}$.*

Proof. Let $S = \{x \in \mathbb{F}_p : x < p/3\}$ and $T = \{x \in \mathbb{F}_p : x > 2p/3\}$. Observe that $2S \cap T = \emptyset$ and $S \cap 2T = \emptyset$, which implies that there is no i such that a_i and a_{i+1} are both 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 $1 \leq i \leq k-1$ and let $a_1' \in [1, p/3)$ be such that $a_1' \equiv a_1 \pmod p$. It follows that $a_i' \equiv a_i \pmod p$ for each i and $a_k' \geq 2^{k-1} \geq p/3$. Now consider the smallest j with $a_j' > p/3$. Then $p/3 \geq a_{j-1}' = a_j'/2$ by the minimality of j, and $p/3 \leq a_j \leq 2p/3$, which is a contradiction. If the sequence is contained in T, then $(-a_1, -a_2, \dots, -a_k)$ is contained in S and using the same argument for this sequence proves the claim. ∎

We now bound the total variation between the distribution of random binary expansions modulo p and the uniform distribution. In Appendix C, we show that Lemma 4.3 is tight assuming there are infinitely many Mersenne primes.

LEMMA 4.3. *There exists an absolute constant $K > 0$ such that the following holds. Let $p > 2$ be prime, let $\epsilon > 0$, and let $t \geq K \cdot c^{-1}(1-c)^{-1} \log(p/\epsilon) \log p$. Define the random variable $Y = \sum_{i=0}^{t} 2^i \cdot Z^{(i)} \in \{0, \dots, 2^{t+1} - 1\}$, and define the random variable $X \in \mathbb{F}_p$ by $X \equiv Y \pmod p$. Then, letting $\mathcal{L}(X)$ denote the law of X, we have*

$$d_{TV}(\mathcal{L}(X), \mathrm{Unif}[\mathbb{F}_p]) \leq \epsilon.$$

Proof. Let $f : \mathbb{F}_p \to \mathbb{R}$ be the probability mass function of X. By definition, we have that

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

Now observe that $\hat{f}(s)$ is given by

$$\begin{aligned} \hat{f}(s) = \sum_{x=0}^{p-1} f(x) \omega^{sx} &= \sum_{z \in \{0,1\}^{t+1}} \left(\prod_{i=0}^{t} q_i^{z_i} (1-q_i)^{1-z_i} \right) \left(\omega^{s \sum_{i=0}^{t} 2^i \cdot z_i} \right) \\ &= \prod_{i=0}^{t} \left(1 - q_i + q_i \cdot \omega^{2^i \cdot s} \right). \end{aligned}$$

The last equality follows directly from expanding the product. 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

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

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

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

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

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

Fix some $s \in \mathbb{F}_p$ with $s \neq 0$. By Claim 4.2, any $\lceil 1 + \log_2(p/3) \rceil$ consecutive terms of the sequence $s, 2s, \dots, 2^t s \in \mathbb{F}_p$ contain an element between $p/3$ and $2p/3$. Therefore, this sequence contains at least $m = \left\lceil \frac{\log(4\epsilon^2/p)}{\log(1-3c(1-c))} \right\rceil$ such terms, which implies that

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

by the inequality above and the fact that each term in this product is at most 1. Since this holds for each $s \neq 0$, it now follows that

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

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

Using the above lemma, we can now prove the main result of this section for $p > 2$. The idea is to rejection sample $Z = (Z^{(0)}, \ldots, Z^{(t)})$ conditioned on $X \equiv x \pmod{p}$.

Proof of Lemma 4.1. Define the random variable $Y = \sum_{i=0}^{t} 2^i \cdot Z^{(i)} \in \{0, \ldots, 2^{t+1}-1\}$, and define the random variable $X \in \mathbb{F}_p$ by $X \equiv Y \pmod{p}$, as in Lemma 4.3. Let $K > 0$ be large enough that, by Lemma 4.3, we have

$$d_{\mathrm{TV}}\left(\mathcal{L}(X), \mathrm{Unif}[\mathbb{F}_p]\right) < \epsilon/(2p).$$

We sample a random variable $\tilde{Y}_x \in \{0, \ldots, 2^{t+1} - 1\}$ by rejection sampling from the distribution $\mathcal{L}(Y)$ until receiving an element congruent to x modulo p or reaching the cutoff of

$$m = \left\lceil \frac{\log(\epsilon/2)}{\log(1 - 1/(2p))} \right\rceil = O\left(p \log(1/\epsilon)\right)$$

rounds, in which case we stop and set $\tilde{Y}_x$ to an arbitrary value congruent to x. We then return $\tilde{Z}_x = (\tilde{Z}_x^{(0)}, \ldots, \tilde{Z}_x^{(t)})$, the binary expansion of $\tilde{Y}_x$ from lowest-order bit to highest-order bit.

By construction, it holds that $\sum_{i=0}^{t} 2^i \cdot \tilde{Z}_x^{(i)} = \tilde{Y}_x \equiv x \pmod{p}$ almost surely. Furthermore, the runtime bound follows because each sample from $\mathcal{L}(Y)$ can be obtained in $O(t)$ time by sampling $Z^{(0)}, Z^{(1)}, \ldots, Z^{(t)}$ and forming the number with binary digits $Z^{(t)}, Z^{(t-1)}, \ldots, Z^{(0)}$. Checking whether this number is congruent to x modulo p takes $O(t \log(p))$ time by Theorem 3.3 of [72].

It remains to prove that $(\tilde{Z}_x^{(0)}, \ldots, \tilde{Z}_x^{(t)})$ is close to $(Z^{(0)}, \ldots, Z^{(t)})$ in total variation if x is chosen uniformly in $\mathbb{F}_p$. We begin by considering the case of fixed $x \in \mathbb{F}_p$. Let Y_x be a random variable with the conditional law $\mathcal{L}(Y_x) \triangleq \mathcal{L}(Y | Y \equiv x \pmod{p})$. If we receive a sample from $\mathcal{L}(Y)$ congruent to x by the mth round of rejection sampling, then it is exactly sampled from $\mathcal{L}(Y_x)$. Therefore, $d_{\mathrm{TV}}(\mathcal{L}(\tilde{Y}_x), \mathcal{L}(Y_x))$ 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

$$\mathbb{P}[X = x] \geq 1/p - d_{\mathrm{TV}}\left(\mathcal{L}(X), \mathrm{Unif}[\mathbb{F}_p]\right) > 1/p - \epsilon/(2p) \geq 1/(2p)$$

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

$$(1 - \mathbb{P}[X = x])^m \leq (1 - 1/(2p))^m \leq \epsilon/2.$$

So we may conclude that, for any fixed $x \in \mathbb{F}_p$,

$$d_{\mathrm{TV}}(\mathcal{L}(\tilde{Y}_x), \mathcal{L}(Y_x)) \leq \epsilon/2.$$

Now let $R \sim \mathrm{Unif}[\mathbb{F}_p]$. By the above inequality, we have

$$d_{\mathrm{TV}}(\mathcal{L}(\tilde{Y}_R), \mathcal{L}(Y_R)) \leq \frac{1}{p} \sum_{x \in \mathbb{F}_p} d_{\mathrm{TV}}(\mathcal{L}(\tilde{Y}_x), \mathcal{L}(Y_x)) \leq \epsilon/2. \tag{4.1}$$

We now bound the total variation distance between $\mathcal{L}(Y_R)$ and $\mathcal{L}(Y)$. Let $X' \sim \mathcal{L}(X)$ be independent of the other variables, and note that $\mathcal{L}(Y_{X'}) = \mathcal{L}(Y)$ since, for any $y \in \{0, \dots, 2^{t+1} - 1\}$, Bayes' rule implies

$$\begin{aligned}\mathbb{P}(Y_{X'} = y) &= \mathbb{P}(Y_{X'} = y \mid Y_{X'} \equiv y \pmod p) \cdot \mathbb{P}(Y_{X'} \equiv y \pmod p) \\ &= \mathbb{P}(Y_{X'} = y \mid X' \equiv y \pmod p) \cdot \mathbb{P}(X' \equiv y \pmod p) \\ &= \mathbb{P}(Y = y \mid Y \equiv y \pmod p) \cdot \mathbb{P}(Y \equiv y \pmod p) \\ &= \mathbb{P}(Y = y).\end{aligned}$$

So by the data processing inequality, since $x \mapsto Y_x$ is a Markov transition sending R to Y_R and X' to $Y_{X'}$,

$$\begin{aligned}d_{\mathrm{TV}}(\mathcal{L}(Y), \mathcal{L}(Y_R)) &= d_{\mathrm{TV}}(\mathcal{L}(Y_{X'}), \mathcal{L}(Y_R)) \\ &\le d_{\mathrm{TV}}(\mathcal{L}(X'), \mathcal{L}(R)) = d_{\mathrm{TV}}(\mathcal{L}(X), \mathrm{Unif}[\mathbb{F}_p]) < \epsilon/2. \end{aligned} \tag{4.2}$$

Finally, since $(\tilde{Z}_R^{(t)}, \dots, \tilde{Z}_R^{(0)})$ is the binary expansion of $\tilde{Y}_R$, and $(Z^{(t)}, \dots, Z^{(0)})$ is the binary expansion of Y, the data processing inequality implies

$$d_{\mathrm{TV}}(\mathcal{L}(\tilde{Z}_R^{(0)}, \dots, \tilde{Z}_R^{(t)}), \mathcal{L}(Z^{(0)}, \dots, Z^{(t)})) \le d_{\mathrm{TV}}(\mathcal{L}(\tilde{Y}_R), \mathcal{L}(Y)). \tag{4.3}$$

We bound the right-hand side of (4.3) with the triangle inequality, (4.1), and (4.2):

$$\begin{aligned}&d_{\mathrm{TV}}(\mathcal{L}(\tilde{Z}_R^{(0)}, \dots, \tilde{Z}_R^{(t)}), \mathcal{L}(Z^{(0)}, \dots, Z^{(t)})) \\ &\qquad \le d_{\mathrm{TV}}(\mathcal{L}(\tilde{Y}_R), \mathcal{L}(Y_R)) + d_{\mathrm{TV}}(\mathcal{L}(Y_R), \mathcal{L}(Y)) < \epsilon/2 + \epsilon/2 = \epsilon.\end{aligned}$$

This completes the proof. ∎

We conclude with a sampling result analogous to Lemma 4.1 but for $p = 2$.

LEMMA 4.4 (sampling lemma for $p = 2$). *There exists a constant $K > 0$ such that the following holds. Let $\epsilon > 0$ and $t \ge Kc^{-1}(1-c)^{-1}\log(1/\epsilon)$. Then there exists an $O(t\log(1/\epsilon))$-time randomized algorithm that, given $x \in \mathbb{F}_2$, outputs $\tilde{Z}_x = (\tilde{Z}_x^{(0)}, \dots, \tilde{Z}_x^{(t)}) \in \{0,1\}^{t+1}$ satisfying $\sum_{i=0}^t \tilde{Z}_x^{(i)} \equiv x \pmod 2$ almost surely. Moreover, if $R \sim \mathrm{Unif}[\mathbb{F}_2]$, then $d_{TV}(\mathcal{L}(\tilde{Z}_R), \mathcal{L}(Z)) < \epsilon$.*

Proof. By induction on t, one may show that

$$\mathbb{P}\left[\sum_{i=0}^t Z^{(i)} \equiv 0 \pmod 2\right] = \frac{1}{2} + \frac{\prod_{i=0}^t (1-2q_i)}{2}.$$

If t satisfies the lower bound $t \ge \lceil \log(\epsilon/4)/\log(|1-2c|)\rceil + 1 = O(c^{-1}(1-c)^{-1}\log(1/\epsilon))$, it holds that $d_{\mathrm{TV}}(\mathcal{L}(\sum_{i=0}^t Z^{(i)} \pmod 2), \mathcal{L}(R)) < \min(1/4, \epsilon/2)$.

The proof now proceeds analogously to the proof of Lemma 4.1. We sample $\tilde{Z}_x = (\tilde{Z}_x^{(0)}, \dots, \tilde{Z}_x^{(t)})$ by rejection sampling from $\mathcal{L}(Z)$ until receiving a vector whose sum is congruent to x modulo 2, or cutting off at $\Theta(\log(1/\epsilon))$ rounds. This takes $O(t\log(1/\epsilon))$ time, because it consists of at most $O(\log(1/\epsilon))$ rounds of sampling fresh copies of $Z^{(i)} \sim \mathrm{Ber}(q_i)$ for all $i \in \{0, \dots, t\}$ and checking whether $\sum_{i=0}^t Z^{(i)} \equiv x \pmod 2$. Let Z_x be a random variable with the conditional law $\mathcal{L}(Z_x) \triangleq \mathcal{L}(Z \mid Z \equiv x \pmod 2)$. Then the rejection sampling outputs $\tilde{Z}_x$ satisfying $d_{\mathrm{TV}}(\mathcal{L}(\tilde{Z}_x), \mathcal{L}(Z_x)) \le \epsilon/2$, so

$$d_{\mathrm{TV}}(\mathcal{L}(\tilde{Z}_R), \mathcal{L}(Z_R)) \le \epsilon/2. \tag{4.4}$$

Further, by applying the data processing inequality with Markov kernel $x \mapsto Z_x$, with reasoning analogous to the proof of (4.2), we derive

$$d_{\mathrm{TV}}(\mathcal{L}(Z_R), \mathcal{L}(Z)) \le d_{\mathrm{TV}}(\mathcal{L}(R), \mathcal{L}(\textstyle\sum_{i=0}^{t} Z^{(i)} \pmod 2))) < \epsilon/2. \tag{4.5}$$

Combining (4.4) and (4.5) with the triangle inequality yields $d_{\mathrm{TV}}(\mathcal{L}(\tilde{Z}_R), \mathcal{L}(Z)) < \epsilon$. ∎

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

5.1. GREEDY-RANDOM-SAMPLING. In this section, we consider a natural greedy algorithm GREEDY-RANDOM-SAMPLING for counting k-cliques in an 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 $\mathrm{CN}_G(A)$ to be

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

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

This algorithm is an extension of the classical greedy algorithm for finding $\log_2 n$ sized cliques in $G(n, 1/2)$ in [54, 45], the Metropolis process examined in [51], and the greedy procedure solving k-CLIQUE on $G(n, c)$ with $c = \Theta\left(n^{-2/(k-1)}\right)$ discussed by Rossman in [71]. 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 [59, 64]. 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 D for completeness.

LEMMA 5.1. *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 $1 - O\big(n^{-\alpha t(1-s^{-1})\binom{\kappa+2}{s-1}}\big)$ for any fixed nonnegative integer t, where the constant in the $O(\cdot)$ notation can depend on t.*

In particular, this implies that the clique number of $G(n,c,s)$ is typically at most $(s!\alpha^{-1})^{\frac{1}{s-1}} + s$. In the graph case of $s = 2$, this simplifies to $2\alpha^{-1} + 2$. 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 postprocessing step. The last subsection gives an alternative deterministic algorithm with a similar performance to GREEDY-RANDOM-SAMPLING.

5.2. Sample complexity and runtime of GREEDY-RANDOM-SAMPLING. In this section, we analyze the runtime of GREEDY-RANDOM-SAMPLING and give 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 $O(kn + k\log n) = O(n)$ time as long as $k = O(1)$. Outputting $|S|$ in GREEDY-RANDOM-SAMPLING therefore yields an $O(nT)$ time algorithm for $\#(k,s)$-CLIQUE on $G(n,c,s)$ that succeeds with high probability.

The following theorem provides 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. Its proof is deferred to Appendix E.

THEOREM 5.2. *Let k and s be constants, and let $c = \Theta(n^{-\alpha})$ for some $\alpha \in (0,1)$. Let τ be the largest integer satisfying $\alpha\binom{\tau}{s-1} < 1$, and suppose that*

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

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 the algorithm* GREEDY-RANDOM-SAMPLING *and over the choice of random hypergraph $G \sim G(n,c,s)$.*

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

COROLLARY 5.3. *Suppose that k and s are constants, and suppose $c = \Theta(n^{-\alpha})$ for some $\alpha \in (0,1)$. Let τ be the largest integer satisfying $\alpha\binom{\tau}{s-1} < 1$. Then the following hold:*

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

By Lemma 5.1, the hypergraph $G \sim G(n, c, s)$ has clique number $\omega(G) \leq \kappa + 2$ with probability $1 - 1/\mathrm{poly}(n)$, where $\kappa \geq s$ is the largest positive integer satisfying $\alpha\binom{\kappa}{s-1} < s$. In particular, when $k > \kappa + 2$ in the theorem above, the algorithm outputting zero succeeds with probability $1 - 1/\mathrm{poly}(n)$ 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 $0 < \alpha \leq s\binom{k-1}{s-1}^{-1}$. In the graph case of $s = 2$, this simplifies to the familiar condition that $0 < \alpha \leq \frac{2}{k-1}$. We also remark that when $k < \tau + 1$, the runtime of this algorithm is an $\tilde{O}(n)$ factor off from the expectation of the quantity being counted, the number of k-cliques in $G \sim G(n, c, s)$.

5.3. Postprocessing 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 [62]. Our improved runtime for the algorithm GREEDY-RANDOM-SAMPLING is stated in the following theorem.

THEOREM 5.4. *Suppose that $k > 2$ is a fixed positive integer and $c = \Theta(n^{-\alpha})$, where $0 < \alpha \leq \frac{2}{k-1}$ is also fixed. Then there exists 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 postprocessing:

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 5.2.
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
$$M_P = \begin{cases} M_1^2 & \text{if } k \equiv 0 \pmod 3, \\ M_1 M_2 & \text{if } k \equiv 1 \pmod 3, \\ M_2 M_3 & \text{if } k \equiv 2 \pmod 3. \end{cases}$$
5. Output the sum of entries
$$\sum_{(A,B) \in \mathcal{S}} (M_P)_{A,B}$$
where $\mathcal{S}$ is the support of M_1 if $k \equiv 0 \pmod 3$ and $\mathcal{S}$ is the support of M_2 if $k \not\equiv 0 \pmod 3$.

We will show that this algorithm solves $\#(k, 2)$-CLIQUE with probability $1 - n^{-\omega(1)}$ when $k \equiv 1 \pmod 3$. The cases when $k \equiv 0, 2 \pmod 3$ follow from a nearly identical argument. By Theorem 5.2, 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 a 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 postprocessing. Observe that the number of cliques output by a call to GREEDY-RANDOM-SAMPLING with T iterations is at most T. Also note that if $\alpha \leq \frac{2}{k-1}$, then $\tau \geq \lfloor \frac{k}{2} \rfloor - 1$. If $k \geq 3$, then it follows that $\tau + 1 \geq \lfloor \frac{k}{2} \rfloor \geq \lceil \frac{k}{3} \rceil$. It follows by Theorem 5.2 that $\max\{|S_1|, |S_2|\} = \tilde{O}\big(n^{\lceil k/3 \rceil + 1 - \alpha\binom{\lceil k/3 \rceil}{2}}\big)$. Note that computing the matrix M_P takes $\tilde{O}\left(\max\{|S_1|, |S_2|\}^\omega\right) = \tilde{O}\big(n^{\omega\lceil k/3 \rceil + \omega - \omega\alpha\binom{\lceil k/3 \rceil}{2}}\big)$ time. Now observe that all other steps of the algorithm run in $\tilde{O}\big(n^{2\lceil k/3 \rceil - 2\alpha\binom{\lceil k/3 \rceil}{2}}\big)$ time, which completes the proof of the theorem since the matrix multiplication constant satisfies $\omega \geq 2$. ∎

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 $k \neq 0 \pmod 3$. 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 $|\mathrm{cl}_k(G)| \leq T$. Theorem 5.2 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 $k \leq \tau + 1$, these upper bounds are a polylog(n) factor from the expectation of $|\mathrm{cl}_k(G)|$. While this was sufficient in the proof of Theorem 5.4, stronger upper bounds will be needed in the next subsection to analyze our deterministic iterative algorithm. 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 the reader to [75, 50, 49, 25] for a survey of the area and recent results. Given a hypergraph H, let $N(n, m, H)$ denote the largest number of copies of H that can be constructed in an s-uniform hypergraph with at most n vertices and m hyperedges. Define the quantity

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

The following large deviations result from [30] generalizes a graph large deviations bound from [49] to hypergraphs to obtain the following result.

THEOREM 5.5 (Theorem 4.1 from [30]). *For every s-uniform hypergraph H and every fixed $\epsilon > 0$, there exists a constant $C(\epsilon, H)$ such that for all $n \geq |V(H)|$ and $c \in (0, 1)$, it holds that*

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

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

Proposition 4.3 in [30] shows that if H is a d-regular s-uniform hypergraph and $c \geq n^{-s/d}$, then $M_H(n,c) = \Theta(n^s c^d)$. This implies that

$$\mathbb{P}\left[|\mathrm{cl}_k(G)| \geq (1+\epsilon)\binom{n}{k} c^{\binom{k}{s}}\right] \leq \exp\left(-C'(\epsilon,s,k)\cdot n^s c^{\binom{k-1}{s-1}}\right) \tag{5.1}$$

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.

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

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

Suppose that C_t are chosen to be any high probability upper bounds on the number of t-cliques in $G \sim G(n,c,s)$ such as the bounds in Theorem 5.5. Then we have the following guarantees for the algorithm IT-GEN-CLIQUES.

THEOREM 5.6. *Suppose that k and s are constants and $c = \Theta(n^{-\alpha})$ for some $\alpha \in (0,1)$. Let τ and κ be the largest integers satisfying $\alpha\binom{\tau}{s-1} < 1$ and $\alpha\binom{\kappa}{s-1} < s$, and let $C_t = 2n^t c^{\binom{t}{s}}$ 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 the following hold:*

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

Proof. Suppose that $k \leq \kappa + 1$. We first show that $S_k = \mathrm{cl}_k(G)$ with probability $1 - n^{-\omega(1)}$ in the algorithm IT-GEN-CLIQUES. By a union bound and (5.1), it follows that $|\mathrm{cl}_t(G)| < C_t$ for each $s \leq t \leq k$ with probability at least $1 - (k-s+1)n^{-\omega(1)}$ since $k \leq \kappa + 1$. 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

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

since $k = O(1)$. To see the second inequality, note that $\log_n(C_{t+1}/C_t) = 1 - \alpha\binom{t}{s-1} + O(1/\log n)$. This implies that $C_{t+1} = \Omega(C_t)$ if $t \leq \tau$ and $C_t = O(C_{\tau+1})$ for all $s \leq t \leq k$. 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, the algorithm

GREEDY-RANDOM-SAMPLING can be modified to match this runtime up to a polylog(n) factor by instead generating the $(k-1)$-cliques of G and applying the last step of IT-GEN-CLIQUES to generate the k-cliques of G. We also remark that IT-GEN-CLIQUES can also be used instead of GREEDY-RANDOM-SAMPLING in step 1 of the algorithm in Theorem 5.4, yielding a nearly identical runtime of $\tilde{O}\big(n^{\omega\lceil k/3\rceil-\omega\alpha\binom{\lceil k/3\rceil-1}{2}}\big)$ for $\#(k,2)$-CLIQUE on inputs sampled from $G(n,c)$.

6. 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 whether the constant C in the exponent of our lower bounds for the graph case of $s=2$ can be improved from 1 to $\omega/9$.

Raising error tolerance for average-case hardness. A natural question is whether the error tolerance of the worst-case to average-case reductions in Theorems 2.8 and 2.9 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 ϵ, 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 $\epsilon \ll \sigma$ for approximation to be hard.

Inhomogeneous Erdős–Rényi hypergraphs. Consider an inhomogeneous Erdős–Rényi hypergraph model, where each hyperedge e is independently chosen to be in the hypergraph with probability $c(e)$. Also suppose that we may bound $c(e)$ uniformly away from 0 and 1 (that is, $c(e) \in [c, 1-c]$ for all possible hyperedges e and for some constant c). We would like to prove that $\#(k,s)$-CLIQUE and PARITY-(k,s)-CLIQUE are hard on average for inhomogeneous Erdős–Rényi hypergraphs. Unfortunately, this does not follow directly from our proof techniques because step 5 in the proof of Theorems 2.8 and 2.9 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 $c(e) \in [c, 1-c]$.

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 2.1.

LEMMA A.1. *Given an algorithm A for* PARITY-(k,s)-CLIQUE *with error probability $< 1/3$ on any s-uniform hypergraph G, there exists 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 nonzero 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 [61]. 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. ∎

Appendix B. Proof of Lemma 3.6. We restate and prove Lemma 3.6.

LEMMA B.1 (Theorem 4 of [41]). *Let $\mathbb{F}$ be a finite field with $|\mathbb{F}| = q$ elements. Let $N > 0$ and $1 \le D < q/12$. Let $f : \mathbb{F}^N \to \mathbb{F}$ be a polynomial of degree at most D. If there exists 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 exists an algorithm B running in time $O((N+D^2)D\log^2 q + T(A,N)\cdot D)$ such that for any *$x \in \mathbb{F}^N$, it holds that $\mathbb{P}[B(x) = f(x)] > 2/3$.*

Proof. Our proof of the lemma is based off of the proof that appears in [8]. The only difference is that in [8], the lemma is stated only for finite fields whose size is a prime. Suppose we wish to calculate $f(x)$ for $x \in \mathbb{F}^N$. In order to do this, choose $y_1, y_2 \overset{i.i.d}{\sim} \mathrm{Unif}[\mathbb{F}^N]$, and define the polynomial $g(t) = x + ty_1 + t^2 y_2$, where $t \in \mathbb{F}$. We use A to evaluate $f(g(t))$ at m different values $t_1, \dots, t_m \in \mathbb{F}$. This takes $O(mN\log^2 q + m\cdot T(A,N))$ time. Suppose without loss of generality that $D \ge 9$. Since $g(t_i)$ and $g(t_j)$ are pairwise independent and uniform in $\mathbb{F}^N$ for any distinct $t_i, t_j \ne 0$, by the second-moment method, with probability $> 2/3$, at most $(m-2D)/2$ of our evaluations of $f(g(t))$ will be incorrect if we take $m = 12D$. Thus, since $f(g(t))$ is a univariate polynomial of degree at most $2D$, we may use Berlekamp–Welch to recover $f(g(0)) = f(x)$ in $O(m^3)$ arithmetic operations over $\mathbb{F}$, each of which takes $O(\log^2 q)$ time. ∎

Appendix C. Tightness of bounds in section 4. In this appendix, we briefly discuss the tightness of the bounds on t in Lemma 4.3 and how the case of $c = 1/2$ differs from $c \ne 1/2$. Note that if $q_i = 1/2$ for each i, then $Y = \sum_{i=0}^{t} Z^{(i)} \cdot 2^i$ is uniformly distributed on $\{0, 1, \dots, 2^{t+1} - 1\}$. It follows that the random variable $X \in \mathbb{F}_p$ defined by $X \equiv Y \pmod p$ satisfies

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

if $0 \le a \le p-1$ is such that $2^{t+1} \equiv a \pmod p$. Here $|\cdot|_+$ denotes $|x|_+ = \max(x, 0)$. Therefore, X 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 following question: Is the additional factor of $\log p$ necessary or an artifact of our analysis? We answer this question with an example suggesting that the extra $\log p$ factor is in fact necessary and that the case $c = 1/2$ is special.

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

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

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

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

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

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

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.

Appendix D. Clique counts in sparse Erdős–Rényi hypergraphs. We prove the following classical lemma from section 5.1.

LEMMA D.1. *For fixed $\alpha \in (0,1)$ and s, let $\kappa \ge s$ 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 $\omega(G) \le \kappa + 1 + t$ with probability at least $1 - O\left(n^{-\alpha t(1-s^{-1})\binom{\kappa+2}{s-1}}\right)$ for any fixed nonnegative integer t, where the constant in the $O(\cdot)$ notation can depend on t.*

Proof. Let $C > 0$ be such that $c \le Cn^{-\alpha}$ for sufficiently large n. For any given set $\{v_1, v_2, \ldots, v_k\}$ of k vertices in $[n]$, the probability that all hyperedges are present among $\{v_1, v_2, \ldots, v_k\}$ 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

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

where we use $\binom{\kappa+1+t}{s-1} \ge \binom{\kappa+1}{s-1} + t\binom{\kappa+1}{s-2}$ by iteratively applying Pascal's identity, as well as $\alpha\binom{\kappa+1}{s-1} > s$ and $k \ge \kappa+2$. Observe that $\kappa = O(1)$ and thus $C^{\binom{k}{s}} = O(1)$. Now, by Markov's inequality, it follows that $\mathbb{P}[\omega(G) \ge k] = \mathbb{P}[|\mathrm{cl}_k(G)| \ge 1] \le \mathbb{E}[|\mathrm{cl}_k(G)|]$, completing the proof of the lemma. ∎

Appendix E. Analysis of GREEDY-RANDOM-SAMPLING. This section is devoted to proving Theorem 5.2, which is restated below for convenience.

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

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

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 the algorithm* GREEDY-RANDOM-SAMPLING *and over the choice of random hypergraph $G \sim G(n,c,s)$.*

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

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

The key property of Z_v is that, in each iteration of GREEDY-RANDOM-SAMPLING, the probability that the k vertices $v_1, v_2, \ldots, v_k$ are chosen in that order is exactly $1/Z_v$. The proof of this theorem will proceed by establishing upper bounds on Z_v that hold for all k-cliques v with high probability over the randomness of G, which will yield a bound on the number of iterations T needed to exhaust all such k-cliques in G.

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

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

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

$\{v_1, v_2, \ldots, v_k\}$ forms a clique in G. For each $i \in \{s-1, s, \ldots, k-1\}$, let $Y_{v,i}$ be the number of common neighbors of $v_1, v_2, \ldots, v_i$ in $V(G)\backslash\{v_1, v_2, \ldots, v_k\}$. Note that $Y_{v,i} \sim \mathrm{Bin}\big(n-k, c^{\binom{i}{s-1}}\big)$ and that $|\mathrm{CN}_G(v_1, v_2, \ldots, v_i)| = k - i + Y_{v,i}$. The standard Chernoff bound for the binomial distribution implies that for all $\delta_i > 0$,

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

Now define κ_i to be

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

for each $i \in \{s-1, s, \ldots, k-1\}$. 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

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

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

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

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

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

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

event

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

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

We now show that as long as B holds over the random choice of G, then the algorithm GREEDY-RANDOM-SAMPLING terminates with $S = \mathrm{cl}_k(G)$ 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 $v = (v_1, v_2, \ldots, v_k)$ of some k-clique $\mathcal{C} = \{v_1, v_2, \ldots, v_k\}$ in G. Recall that in any one of the T iterations of GREEDY-RANDOM-SAMPLING, the probability that the k vertices $v_1, v_2, \ldots, v_k$ are chosen in that order is exactly $1/Z_v$. Since the T iterations of GREEDY-RANDOM-SAMPLING are independent, we have that

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

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

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

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

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

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

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

Acknowledgments. We thank Oded Goldreich and the anonymous reviewers for helpful feedback that greatly improved the exposition. We also thank Frederic Koehler, Dheeraj Nagaraj, and Austin Stromme for inspiring discussions on related topics.

REFERENCES

[1] M. Ajtai, *Generating hard instances of lattice problems*, in Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, 1996, pp. 99–108.

[2] N. Alon and R. B. Boppana, *The monotone circuit complexity of Boolean functions*, Combinatorica, 7 (1987), pp. 1–22.

[3] N. Alon, M. Krivelevich, and B. Sudakov, *Finding a large hidden clique in a random graph*, Random Structures Algorithms, 13 (1998), pp. 457–466.

[4] 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 J. Comput., 35 (2005), pp. 201–216, https://doi.org/10.1137/S0097539701396959.

[5] B. P. Ames and S. A. Vavasis, *Nuclear norm minimization for the planted clique and biclique problems*, Math. Program., 129 (2011), pp. 69–89.

[6] 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, 2018, pp. 866–877.

[7] E. Bach, J. Driscoll, and J. Shallit, *Factor refinement*, J. Algorithms, 15 (1993), pp. 199–222.

[8] 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, 2017, pp. 483–496.

[9] 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 Proceedings of the 57th Annual IEEE Symposium on Foundations of Computer Science, 2016, pp. 428–437.

[10] Q. Berthet and P. Rigollet, *Complexity Theoretic Lower Bounds for Sparse Principal Component Detection*, in Proceedings of the 26th Annual Conference on Learning Theory, 2013, pp. 1046–1066.

[11] A. Bogdanov and L. Trevisan, *Average-case complexity*, Found. Trends Theor. Comput. Sci., 2 (2006), pp. 1–106.

[12] A. Bogdanov and L. Trevisan, *On worst-case to average-case reductions for NP problems*, SIAM J. Comput., 36 (2006), pp. 1119–1159, https://doi.org/10.1137/S0097539705446974.

[13] B. Bollobás and O. Riordan, *Clique percolation*, Random Structures Algorithms, 35 (2009), pp. 294–322.

[14] M. Brennan and G. Bresler, *Optimal average-case reductions to sparse PCA: From weak assumptions to strong hardness*, in Proceedings of the 32nd Annual Conference on Learning Theory, 2019, pp. 469–470.

[15] M. Brennan, G. Bresler, and W. Huleihel, *Reducibility and computational lower bounds for problems with planted sparse structure*, in Proceedings of the 31st Annual Conference on Learning Theory, 2018, pp. 48–166.

[16] M. Brennan, G. Bresler, and W. Huleihel, *Universality of computational lower bounds for submatrix detection*, in Proceedings of the 32nd Annual Conference on Learning Theory, 2019, pp. 417–468.

[17] J.-Y. Cai, A. Pavan, and D. Sivakumar, *On the hardness of permanent*, in Proceedings of the 16th Annual Symposium on Theoretical Aspects of Computer Science, Springer, Berlin, 1999, pp. 90–99.

[18] C. Calabro, R. Impagliazzo, V. Kabanets, and R. Paturi, *The complexity of unique k-SAT: An isolation lemma for k-CNFs*, J. Comput. System Sci., 74 (2008), pp. 386–393.

[19] J. Chen, X. Huang, I. A. Kanj, and G. Xia, *Strong computational lower bounds via parameterized complexity*, J. Comput. System Sci., 72 (2006), pp. 1346–1367.

[20] Y. Chen, *Incoherence-optimal matrix completion*, IEEE Trans. Inform. Theory, 61 (2015), pp. 2909–2923.

[21] Y. Chen and J. Xu, *Statistical-computational tradeoffs in planted problems and submatrix localization with a growing number of clusters and submatrices*, J. Mach. Learn. Res., 17 (2016), 27.

[22] A. Coja-Oghlan and C. Efthymiou, *On independent sets in random graphs*, Random Structures Algorithms, 47 (2015), pp. 436–486.

[23] M. Dalirrooyfard, A. Lincoln, and V. V. Williams, *New techniques for proving fine-grained average-case hardness*, in Proceedings of the 61st Annual IEEE Symposium on Foundations of Computer Science, 2020, pp. 774–785.
[24] Y. Dekel, O. Gurel-Gurevich, and Y. Peres, *Finding hidden cliques in linear time with high probability*, Combin. Probab. Comput., 23 (2014), pp. 29–49.
[25] R. DeMarco and J. Kahn, *Tight upper tail bounds for cliques*, Random Structures Algorithms, 41 (2012), pp. 469–487.
[26] I. Derényi, G. Palla, and T. Vicsek, *Clique percolation in random networks*, Phys. Rev. Lett., 94 (2005), 160202.
[27] Y. Deshpande and A. Montanari, *Finding hidden cliques of size* $\sqrt{N/e}$ *in nearly linear time*, Found. Comput. Math., 15 (2015), pp. 1069–1128.
[28] S. N. Dorogovtsev, A. V. Goltsev, and J. F. Mendes, *Critical phenomena in complex networks*, Rev. Mod. Phys., 80 (2008), pp. 1275–1336.
[29] R. G. Downey and M. R. Fellows, *Fixed-parameter tractability and completeness* II*: On completeness for W*[1], Theoret. Comput. Sci., 141 (1995), pp. 109–131.
[30] A. Dudek, J. Polcyn, and A. Ruciński, *Subhypergraph counts in extremal and random hypergraphs and the fractional q-independence*, J. Combin. Optim., 19 (2010), pp. 184–199.
[31] U. Feige, D. Gamarnik, J. Neeman, M. Z. Rácz, and P. Tetali, *Finding cliques using few probes*, Random Structures Algorithms, 56 (2020), pp. 142–153.
[32] U. Feige and R. Krauthgamer, *Finding and certifying a large hidden clique in a semirandom graph*, Random Structures Algorithms, 16 (2000), pp. 195–208.
[33] 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, 1992, pp. 643–654.
[34] U. Feige and D. Ron, *Finding hidden cliques in linear time*, in Proceedings of the 21st International Meeting on Probabilistic, Combinatorial, and Asymptotic Methods in the Analysis of Algorithms (AofA'10), Discrete Math. Theor. Comput. Sci. Proc., 2010, pp. 189–203.
[35] J. Feigenbaum and L. Fortnow, *Random-self-reducibility of complete sets*, SIAM J. Comput., 22 (1993), pp. 994–1005, https://doi.org/10.1137/0222061.
[36] 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, 2013, pp. 655–664.
[37] D. Gamarnik and E. C. Kizildağ, *Computing the partition function of the Sherrington-Kirkpatrick model is hard on average*, in Proceedings of the IEEE International Symposium on Information Theory, 2020, pp. 2837–2842.
[38] D. Gamarnik and M. Sudan, *Limits of local algorithms over sparse random graphs*, in Proceedings of the 5th ACM Conference on Innovations in Theoretical Computer Science, 2014, pp. 369–376.
[39] S. Gao, *Normal Bases over Finite Fields*, Ph.D. thesis, University of Waterloo, Waterloo, ON, Canada, 1993.
[40] 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, 1991, pp. 33–42.
[41] P. Gemmell and M. Sudan, *Highly resilient correctors for polynomials*, Inform. Process. Lett., 43 (1992), pp. 169–174.
[42] O. Goldreich, *On counting t-cliques mod* 2, in Electronic Colloquium on Computational Complexity (ECCC), 2020, pp. 20–104.
[43] O. Goldreich and G. Rothblum, *Counting t-cliques: Worst-case to average-case reductions and direct interactive proof systems*, in Proceedings of the 59th Annual IEEE Symposium on Foundations of Computer Science, 2018, pp. 77–88.
[44] O. Goldreich and G. N. Rothblum, *Worst-case to average-case reductions for subclasses of P*, in Computational Complexity and Property Testing, Springer, Cham, 2020, pp. 249–295.
[45] G. R. Grimmett and C. J. McDiarmid, *On colouring random graphs*, Math. Proc. Cambridge Philos. Soc., 77 (1975), pp. 313–324.
[46] B. E. Hajek, Y. Wu, and J. Xu, *Computational Lower Bounds for Community Detection on Random Graphs*, in Proceedings of the 28th Annual Conference on Learning Theory, 2015, pp. 899–928.
[47] S. Hirahara and N. Shimizu, *Nearly optimal average-case complexity of counting bicliques under SETH*, in Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA), SIAM, Philadelphia, 2021, pp. 2346–2365, https://doi.org/10.1137/1.9781611976465.140.
[48] A. Itai and M. Rodeh, *Finding a minimum circuit in a graph*, SIAM J. Comput., 7 (1978),

pp. 413–423, https://doi.org/10.1137/0207033.
[49] S. Janson, K. Oleszkiewicz, and A. Ruciński, *Upper tails for subgraph counts in random graphs*, Israel J. Math., 142 (2004), pp. 61–92.
[50] S. Janson and A. Ruciński, *The infamous upper tail*, Random Structures Algorithms, 20 (2002), pp. 317–342.
[51] M. Jerrum, *Large cliques elude the Metropolis process*, Random Structures Algorithms, 3 (1992), pp. 347–359.
[52] A. Juels and M. Peinado, *Hiding cliques for cryptographic security*, Des. Codes Cryptogr., 20 (2000), pp. 269–280.
[53] R. M. Karp, *Reducibility among combinatorial problems*, in Complexity of Computer Computations, Plenum, New York, 1972, pp. 85–103.
[54] R. M. Karp, *Probabilistic analysis of some combinatorial search problems*, in Algorithms and Complexity: New Directions and Recent Results, Academic Press, New York, 1976.
[55] P. Koiran and A. Zouzias, *Hidden cliques and the certification of the restricted isometry property*, IEEE Trans. Inform. Theory, 60 (2014), pp. 4999–5006.
[56] L. Kučera, *Expected complexity of graph partitioning problems*, Discrete Appl. Math., 57 (1995), pp. 193–212.
[57] R. J. Lipton, *New directions in testing*, in Distributed Computing and Cryptography, DIMACS Ser. Discrete Math. Theoret. Comput. Sci. 2, AMS, Providence, RI, 1991, pp. 191–202.
[58] Z. Ma and Y. Wu, *Computational barriers in minimax submatrix detection*, Ann. Statist., 43 (2015), pp. 1089–1116.
[59] C. McDiarmid, *Colouring random graphs*, Ann. Oper. Res., 1 (1984), pp. 183–200.
[60] F. McSherry, *Spectral partitioning of random graphs*, in Proceedings of the 42nd Annual IEEE Symposium on Foundations of Computer Science, 2001, pp. 529–537.
[61] D. E. Muller, *Application of Boolean algebra to switching circuit design and to error detection*, Trans. IRE Prof. Group Electron. Comput., 3 (1954), pp. 6–12.
[62] J. Nešetřil and S. Poljak, *On the complexity of the subgraph problem*, Comment. Math. Univ. Carolin., 26 (1985), pp. 415–419.
[63] G. Palla, I. Derényi, and T. Vicsek, *The critical point of k-clique percolation in the Erdős–Rényi graph*, J. Stat. Phys., 128 (2007), pp. 219–227.
[64] B. Pittel, *On the probable behaviour of some algorithms for finding the stability number of a graph*, Math. Proc. Cambridge Philos. Soc., 92 (1982), pp. 511–526.
[65] M. Rahman and B. Virag, *Local algorithms for independent sets are half-optimal*, Ann. Probab., 45 (2017), pp. 1543–1577.
[66] A. A. Razborov, *Lower bounds for the monotone complexity of some Boolean functions*, Soviet Math. Dokl., 31 (1985), pp. 354–357.
[67] O. Regev, *On lattices, learning with errors, random linear codes, and cryptography*, J. ACM, 56 (2009), 34.
[68] O. Regev, *The learning with errors problem*, in Proceedings of the 25th Annual IEEE Conference on Computational Complexity, 2010, pp. 191–204.
[69] B. Rossman, *On the constant-depth complexity of k-clique*, in Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing, 2008, pp. 721–730.
[70] B. Rossman, *The monotone complexity of k-clique on random graphs*, in Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science, 2010, pp. 193–201.
[71] B. Rossman, *Lower bounds for subgraph isomorphism*, in Proceedings of the International Congress of Mathematicians, World Scientific, Hackensack, NJ, 2018, pp. 3425–3446.
[72] V. Shoup, *A Computational Introduction to Number Theory and Algebra*, Cambridge University Press, Cambridge, UK, 2009.
[73] M. Sudan, *Decoding of Reed Solomon codes beyond the error-correction bound*, J. Complexity, 13 (1997), pp. 180–193.
[74] L. G. Valiant, *The complexity of enumeration and reliability problems*, SIAM J. Comput., 8 (1979), pp. 410–421, https://doi.org/10.1137/0208032.
[75] V. H. Vu, *A large deviation result on the number of small subgraphs of a random graph*, Combin. Probab. Comput., 10 (2001), pp. 79–94.
[76] R. Yuster, *Finding and counting cliques and independent sets in r-uniform hypergraphs*, Inform. Process. Lett., 99 (2006), pp. 130–134.