Edge Estimation with Independent Set Oracles[*]

Paul Beame[†1], Sariel Har-Peled[‡2], Sivaramakrishnan Natarajan Ramamoorthy[§3], Cyrus Rashtchian[¶4], and Makrand Sinha[5]

1 **Paul G. Allen School of Computer Science & Engineering, University of Washington, Seattle, USA**
beame@cs.washington.edu
2 **Dept. of Computer Science, University of Illinois, Urbana-Champaign, USA**
sariel@illinois.edu
3 **Paul G. Allen School of Computer Science & Engineering, University of Washington, Seattle, USA**
sivanr@cs.washington.edu
4 **Paul G. Allen School of Computer Science & Engineering, University of Washington, Seattle, USA**
cyrash@cs.washington.edu
5 **Paul G. Allen School of Computer Science & Engineering, University of Washington, Seattle, USA**
makrand@cs.washington.edu

Abstract

We study the problem of estimating the number of edges in a graph with access to only an independent set oracle. Independent set queries draw motivation from group testing and have applications to the complexity of decision versus counting problems. We give two algorithms to estimate the number of edges in an n-vertex graph: one that uses only polylog(n) bipartite independent set queries, and another one that uses $n^{2/3} \cdot$ polylog(n) independent set queries.

1998 ACM Subject Classification F.1.1 Models of Computation, F.2 Analysis of Algorithms and Problem Complexity

Keywords and phrases Approximate Counting, Independent Set Queries, Sparsification, Importance Sampling

Digital Object Identifier 10.4230/LIPIcs.ITCS.2018.38

1 Introduction

We study the problem of estimating the number of edges in a simple, unweighted, undirected graph $G = ([n], E)$, where $[n] := \{1, 2, \ldots, n\}$ and $m = |E|$, using only an oracle that answers independent set queries. For a parameter $\varepsilon > 0$, we wish to output an estimate $\widetilde{m}$ satisfying $(1 - \varepsilon)m \leq \widetilde{m} \leq (1 + \varepsilon)m$ with high probability. We consider randomized algorithms with access to one of the two following independent set oracles:

[*] A full version of the paper is available at [2], https://arxiv.org/abs/1711.07567
[†] Supported in part by the NSF under agreement CCF-1524246.
[‡] Supported in part by NSF AF awards CCF-1421231 and CCF-1217462. Work done while visiting University of Washington on Sabbatical in 2017.
[§] Supported by the NSF under agreements CCF-1149637, CCF-1420268 and CCF-1524251.
[¶] Work partially completed while the author was at Microsoft Research.

Bipartite independent set (BIS) oracle: Given disjoint subsets $A, B \subseteq [n]$, a BIS query answers whether A, B satisfy $e(A, B) = 0$, where $e(A, B)$ denotes the number of edges with one endpoint in A and the other in B.

Independent set (IS) oracle: Given a subset $A \subseteq [n]$, an IS query answers whether A satisfies $e(A) = 0$, where $e(A)$ denotes the number of edges with both endpoints in A.

Previous work on graph parameter estimation has primarily focused on *local* queries, such as *degree* queries (which output the degree of a vertex v), *edge existence* queries (which answer whether a pair (u, v) forms an edge), or *neighbor* queries (which provide the i^{th} neighbor of a vertex v). However, such queries cannot achieve sub-polynomial query costs on certain lower bound graphs identified by Feige [13] and Goldreich and Ron [15], essentially due to the fact that these queries can only obtain *local* information about the graph. This motivates an investigation of other types of queries that may enable very efficient parameter estimation. The independent set queries described above naturally generalize an edge existence query, and their non-locality opens the door for sub-polynomial query algorithms for various graph parameter estimation tasks.

1.1 Motivation and Related Work

The most relevant previous work comes from the area of sub-linear time algorithms for graph parameter estimation. BIS and IS queries also have interesting connections to the classical area of group testing, to emptiness versus counting questions in computational geometry, and to the complexity of decision versus counting problems.

Graph Parameter Estimation. Many researchers have studied the problem of estimating various parameters of graphs using many types of queries. Feige [13] estimated the number of edges in a graph using degree queries, where a degree query returns the degree $\deg(v)$ of a specified vertex v in $G([n], E)$. In a follow-up work, Goldreich and Ron [15] estimated the number of edges in a graph using both degree and neighbor queries, where a neighbor query returns the j^{th} neighbor of a vertex v for $j, v \in [n]$. Related work has also appeared on estimating the number of stars [16], the minimum vertex cover [18], the number of triangles [10, 20], and the number of k-cliques [9]. A special case of BIS query termed a *group* query (where one of the bipartition sets is a singleton) was considered for testing k-colorability of graphs [3] and edge estimation [24].

The results of Feige [13] and Goldreich and Ron [15] on estimating the number of edges in a graph are quite relevant to our work. Feige [13] showed how to use $O\left(\sqrt{n}/\varepsilon\right)$ degree queries to output an estimate $\widetilde{m}$ that satisfies $(2 - \varepsilon)m \le \widetilde{m} \le (2 + \varepsilon)m$. Moreover, he showed that any algorithm achieving better than a 2-approximation must use a nearly linear number of queries in the worst case. Goldreich and Ron [15] showed that by using both degree and neighbor queries, the approximation could be improved to $(1 - \varepsilon)m \le \widetilde{m} \le (1 + \varepsilon)m$ by using $\sqrt{n} \cdot \operatorname{poly}(\log n, 1/\varepsilon)$ queries. Finally, we mention that Feige [13] and Goldreich and Ron [15] have identified certain hard instances showing that these upper bounds cannot be improved, up to $\operatorname{polylog}(n)$ factors.

Group Testing. A classic estimation problem involves efficiently approximating the number of defective items or infected individuals in a certain collection or population [5, 7, 23]. To query a population, a small group is formed, and all the individuals in the group are tested in one shot. For example, in genome-wide association studies, combined pools of DNA may be tested as a group for certain variants [17]. In group testing, the result of a test often indicates only whether there is at least one infected or defective unit, or if there is none. Such a dichotomous outcome resembles the independent set queries that we study. Group testing in the graph setting suggests the interpretation that we wish to test pairwise interactions between items or individuals, instead of singular events.

Computational Geometry. Certain geometric applications exhibit the phenomena that emptiness queries have more efficient algorithms than counting queries. For example, in three dimensions, for a set P of n points, half-space counting queries (i.e., what is the size of the set $|P \cap h|$, for a query half-space h), can be answered in $O(n^{2/3})$ time, after near-linear time preprocessing. On the other hand, emptiness queries (i.e., is the set $P \cap h$ empty?) can be answered in $O(\log n)$ time. Aronov and Har-Peled [1] used this to show how to answer approximate counting queries (i.e., estimating $|P \cap h|$), with polylogarithmic emptiness queries.

As another geometric example, consider the task of counting edges in disk intersection graphs using GPUs [14]. For these graphs, IS queries decide if a subset of the disks have any intersection (this can be done using sweeping in $O(n \log n)$ time [4]). Using a GPU, one could quickly draw the disks and check if the sets share a common pixel. In cases like this – when IS and BIS oracles have fast implementations – algorithms exploiting independent set queries may be useful.

Decision versus Counting Complexity. A generalization of IS and BIS queries has previously appeared in a line of work investigating the relationship between decision and counting problems [21, 22, 6]. Stockmeyer [21, 22] showed how to estimate the number of satisfying assignments for a given circuit with queries to an NP oracle. Ron and Tsur [19] observed that Stockmeyer implicitly provided an algorithm for estimating set cardinality using *subset* queries, where a subset query specifies a subset $X \subseteq \mathcal{U}$ and answers whether $|X \cap S| = 0$ or not. Subset queries generalize IS and BIS queries because S corresponds to the set of edges in the graph with $|S|$ and X is an arbitrary subset of pairs of vertices.

In what follows, we consider subset queries in the context of edge estimation and fix $|S| = m$ and $|\mathcal{U}| = \binom{n}{2}$. Stockmeyer provided an algorithm using $O(\log\log m \cdot \mathrm{poly}(1/\varepsilon))$ subset queries to estimate m within a factor of $(1+\varepsilon)$ with a constant success probability. Note that for a high probability bound, which is what we focus on in this paper, the algorithm would naively require $O(\log n \cdot \log\log m \cdot \mathrm{poly}(1/\varepsilon))$ queries to achieve success probability at least $1 - 1/n$. Falahatgar, Jafarpour, Orlitsky, Pichapati, and Suresh [12] gave an improved algorithm that estimates m up to a factor of $(1+\varepsilon)$ with probability $1-\delta$ using $2\log\log m + O\big((1/\varepsilon^2)\log(1/\delta)\big)$ subset queries. Nearly matching lower bounds are also known for subset queries [21, 22, 19, 12]. Ron and Tsur [19] also study a restriction of subset queries, called *interval queries*, where the universe $\mathcal{U}$ is ordered and the subsets are intervals of elements. We view the independent set queries as another natural restriction of subset queries.

Analogous to Stockmeyer's results, a recent work of Dell and Lapinskas [6] provides a framework that relates edge estimation using BIS and edge existence queries to a question in fine-grained complexity. They study the relationship between decision and counting versions of problems such as 3SUM and Orthogonal Vectors. We first describe their edge estimation result and then explain their connection to fine-grained complexity. They prove the following for bipartite graphs.

▶ **Theorem 1** ([6]). *For every $0 \le \varepsilon \le 1$, there exists an algorithm using $\frac{O(\log^6 n)}{\varepsilon^2}$ BIS queries and $\frac{n \cdot \mathrm{polylog}(n)}{\varepsilon^4}$ edge existence queries, and outputs $\widetilde{m}$ satisfying $(1-\varepsilon)m \le \widetilde{m} \le (1+\varepsilon)m$ with probability at least $1 - 1/n^2$.*

Dell and Lapinskas [6] used their edge estimation algorithm to obtain approximate counting algorithms for problems in fine-grained complexity. For instance, given an algorithm for 3SUM with runtime T, they obtain an algorithm that estimates the number of YES instances of 3SUM with runtime $T \cdot \frac{O(\log^6 n)}{\varepsilon^2} + \frac{n \cdot \mathrm{polylog}(n)}{\varepsilon^4}$. The relationship is quite simple and natural. The decision version of 3SUM corresponds to checking if there is at least one edge in a certain bipartite graph. The counting version then corresponds to counting the edges in this graph. We note that in their application, the large number $O(n \cdot \mathrm{polylog}(n))$ of edge existence queries does not affect the dominating term in the overall time in their reduction; the larger term in the time is a product of the time to decide 3SUM and the number of BIS queries.

1.2 Our Results

We present two new algorithms. In what follows, let $G = ([n], E)$ be a simple, unweighted graph with $|E| = m$ edges. Our first algorithm, using BIS queries, gives the following.

▶ **Theorem 2.** *Given $n \geq 16$ and $\frac{36 \log n}{\sqrt{m}} \leq \varepsilon \leq \frac{1}{2}$, there exists an algorithm that makes $\frac{\mathrm{polylog}(n)}{\varepsilon^4}$ BIS queries and outputs $\widetilde{m}$ satisfying $(1-\varepsilon)m \leq \widetilde{m} \leq (1+\varepsilon)m$ with probability at least $1 - \frac{\mathrm{polylog}(n)}{n^2}$.*

We remark that since polylog(n) BIS queries can simulate a degree query (see the full version [2] for a proof) one obtains a $(2 \pm \varepsilon)$-factor approximation of m by using Feige's algorithm [13], which uses degree queries. This algorithm uses $O(\sqrt{n} \cdot \mathrm{polylog}(n)/\mathrm{poly}(\varepsilon))$ BIS queries. Theorem 2, however, provides better guarantees in terms of the approximation and the number of BIS queries.

Compared to the result of Dell and Lapinskas [6] (Theorem 1), our algorithm uses a significantly fewer number of queries, since we do not have to make $n \cdot \mathrm{polylog}(n)$ edge existence queries. In terms of their specific applications, it does not seem that our improvement implies anything significant for fine-grained complexity. It would be interesting to find problems where a more efficient BIS estimation algorithm would lead to better decision versus counting complexity results.

Our second algorithm, using only IS queries, gives the following.

▶ **Theorem 3.** *Given $n \geq 8$ and $\frac{324 \log^4 n}{\sqrt{m}} \leq \varepsilon \leq \frac{1}{2}$, there exists an algorithm that makes $\min\left\{\frac{n^2}{\varepsilon^2 m}, \frac{\sqrt{m}}{\varepsilon}\right\} \cdot \mathrm{polylog}(n)$ IS queries and outputs $\widetilde{m}$ satisfying $(1-\varepsilon)m \leq \widetilde{m} \leq (1+\varepsilon)m$ with probability at least $1 - \frac{1}{n^2}$.*

The first term $\frac{n^2}{\varepsilon^2 m} \cdot \mathrm{polylog}(n)$ comes from a folklore algorithm that estimates the number of edges using edge existence queries (see for a proof). The second term $\frac{\sqrt{m}}{\varepsilon} \cdot \mathrm{polylog}(n)$ is the number of queries used by our new algorithm.

Since $\min\left\{\frac{n^2}{\varepsilon^2 m}, \frac{\sqrt{m}}{\varepsilon}\right\} \leq \frac{n^{2/3}}{\varepsilon^{4/3}}$, we also get the following corollary.

▶ **Corollary 4.** *Given $n \geq 8$ and $\frac{324 \log^4 n}{\sqrt{m}} \leq \varepsilon \leq \frac{1}{2}$, there is an algorithm that makes $\frac{n^{2/3}}{\varepsilon^{4/3}} \cdot \mathrm{polylog}(n)$ IS queries and outputs $\widetilde{m}$ satisfying $(1-\varepsilon)m \leq \widetilde{m} \leq (1+\varepsilon)m$ such that with probability at least $1 - \frac{1}{n^2}$.*

Comparing the above theorems, we observe that, perhaps surprisingly, BIS queries are much more effective for estimating the number of edges than IS queries.

Table 1 Comparison of the best known algorithms using a variety of queries for estimating the number of edges m in a graph with n vertices. The bounds stated are for high probability results, with error probability at most $1/n$. Constant factors are suppressed for readability.

Query Types	Approximation	# Queries (up to const. factors)	Reference
Edge Existence	$1+\varepsilon$	$\frac{n^2}{m} \cdot \text{poly}(\log n, 1/\varepsilon)$	Folklore (see [2])
Degree	$2+\varepsilon$	$\frac{\sqrt{n}}{\varepsilon} \cdot \log n$	[13]
Degree + Neighbor	$1+\varepsilon$	$\sqrt{n} \cdot \text{poly}(\log n, 1/\varepsilon)$	[15]
Subset	$1+\varepsilon$	$\log n \cdot \text{poly}(1/\varepsilon)$	[22, 12]
BIS + Edge Existence	$1+\varepsilon$	$n \cdot \text{poly}(\log n, 1/\varepsilon)$	[6]
BIS	$1+\varepsilon$	$\text{poly}(\log n, 1/\varepsilon)$	This Work
IS	$1+\varepsilon$	$\min\left\{\sqrt{m}, \frac{n^2}{m}\right\} \cdot \text{poly}(\log n, 1/\varepsilon)$	This Work

1.2.1 Comparison with Other Queries.

Table 1 quantitatively summarizes the results for estimating the number of edges in a graph in the context of various query types. Given some of the results in Table 1 on edge estimation using other types of queries, a natural question is how well BIS and IS queries can simulate such queries. In the full version [2] of the paper, we show that polylog(n) BIS queries are sufficient to simulate degree queries. On the other hand, we do not know how to simulate a neighbor query (to find a specific neighbor) with few BIS queries, but a random neighbor of a vertex can be found with $O(\log n)$ BIS queries (see [3]). For IS queries, it turns out that estimating the degree of a vertex v up to a constant factor requires at least $\Omega\left(\frac{n}{\deg(v)}\right)$ IS queries (we expand on this in the full version of the paper).

Notation. Throughout this text, log and ln will denote the logarithm taken in base two and e, respectively. For a positive integer k, the set $\{1, \ldots, k\}$ will be denoted by $[k]$. The notation $x = \text{polylog}(n)$ means $x = O(\log^c n)$ for some constant $c > 0$. When we say $A_1, \ldots, A_k$ is a *partition* of the set A into k parts we allow A_i to be empty. In particular, a uniformly random partition of A into k parts is chosen by coloring each element of A with a random number in $[k]$ and identifying A_i with elements colored i.

2 Overview of the Algorithms

We describe our algorithms using BIS and IS queries separately.

2.1 BIS Algorithm

Our discussion of the BIS algorithm will parallel Figure 1, which depicts the main components of one level of our recursive algorithm.

Our algorithms rely on the ability to exactly count the edges between two subsets of vertices, in time nearly linear in the number of such edges. In particular, we provide a simple,

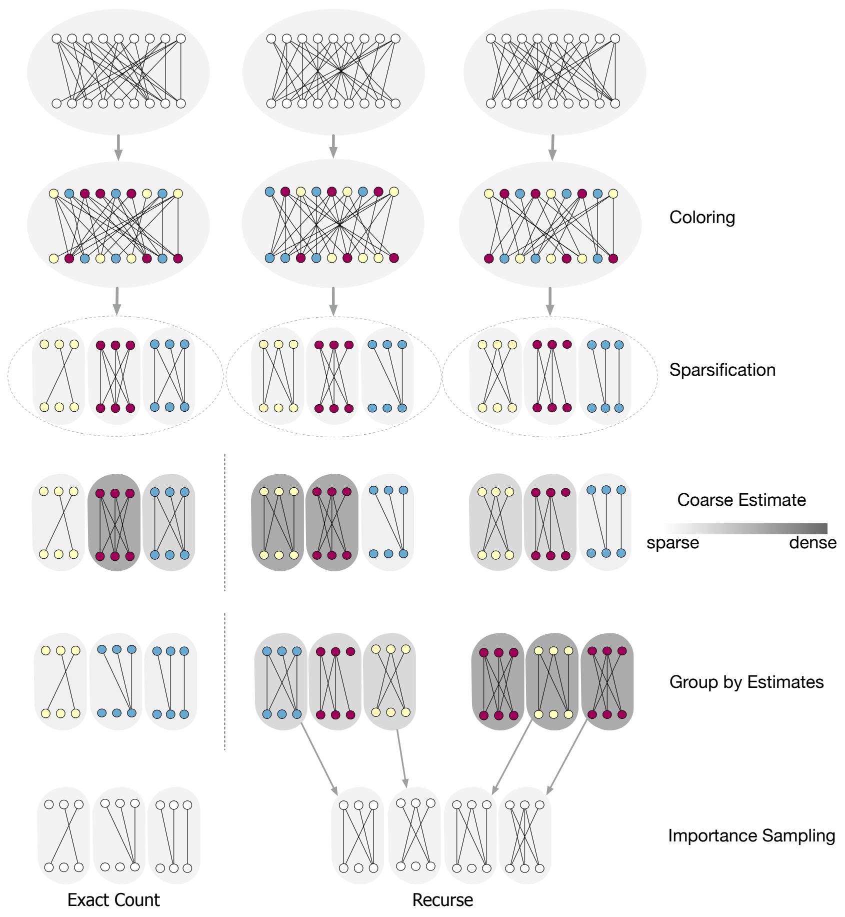

Figure 1 A depiction of one level of the BIS algorithm. In the first step, we color the vertices and sparsify the graph by only looking at the edges between vertices of the same color. In the second step, we coarsely estimate the number of edges in each colored subgraph. Next, we group these subgraphs based on their coarse estimates, and we subsample from the groups with a relatively large number of edges. In the final step, we exactly count the edges in the sparse subgraphs, and we recurse on the dense subgraphs.

deterministic divide and conquer algorithm to determine $e(A, B)$ using $O(e(A, B) \log n)$ BIS queries. More concretely, we will prove the following in the next section.

▶ **Lemma 5.** *For disjoint $A, B \subseteq [n]$, there is a deterministic algorithm that exactly computes $e(A, B)$ using at least $\frac{e(A,B)+1}{2}$ and at most $5 \cdot e(A, B)\lceil \log n \rceil + 1$ BIS queries.*

Given this algorithm, it is natural to wonder if the graph could be sparsified in such a way that the number of remaining edges is a good estimate for the original number of edges (after scaling). Consider sparsifying the graph by coloring the vertices of graph and only looking at the edges going between certain pairs of color classes (in our algorithm, these pairs will be a matching of the color classes). We prove that it suffices to only count the edges between the color classes, and we can ignore the edges with both endpoints inside a single color class.

▶ **Lemma 6** (Basic Sparsification). *Let G be an n-vertex graph with m edges. For k such that $1 \leq k \leq \lfloor n/2 \rfloor$, let $A_1, \ldots, A_k, B_1, \ldots, B_k$ be a uniformly random partition of $[n]$. Then,*

$$\mathbb{P}\left[\left|2k\sum_{i=1}^{k} e(A_i, B_i) - m\right| \geq 18k \cdot \sqrt{m}\log n\right] \leq \frac{1}{n^4}.$$

An important consequence of this lemma is that we can assume without loss of generality that the graph is bipartite. Indeed, invoking the lemma with $k = 2$, estimating the edges between the two color classes is equivalent to estimating the total number of edges, up to a factor of two. In what follows, we consider colorings that respect the bipartition.

After coloring the graph, we have reduced the problem to estimating the total number of edges in a collection of bipartite subgraphs. However, certain subgraphs may still have a large number of edges, and it would be too expensive to directly use the exact counting algorithm. To remedy this, we develop an algorithm that coarsely estimates the number of edges in a subgraph, up a $O(\log^2 n)$ factor, using only $O(\log^3 n)$ BIS queries.

Using the coarse estimates we can form $O(\log n)$ groups of bipartite subgraphs, where each group contains subgraphs with a comparable number of edges. For the groups with only a polylogarithmic number of edges, we can exactly count edges using polylog(n) BIS queries. For the remaining groups, we subsample a polylogarithmic number of subgraphs from each group. Since the groups contained subgraphs with a similar number of edges, the number of edges in the subsampled subgraphs will be proportional to the total number of edges in the group, up to a scaling factor depending on the group, with high probability. This corresponds to the technique of *importance sampling* that is used for variance reduction when estimating a sum of random variables that have comparable magnitudes.

After exactly counting the edges in the sparse subgraphs, we are left with the task of estimating the number of edges in the subsampled subgraphs. By the sparsification guarantee, the number of edges in these subgraphs has gone down by a factor of k (which will be a constant) with high probability. We now recurse on the collection of subsampled subgraphs. Since the number of edges has gone down by a constant fraction, we only need to repeat this process a logarithmic number of times. Overall, at every level of the recursion, we work with a polylogarithmic number of subgraphs, and hence, we only make a polylogarithmic number of BIS queries in total.

Note that we have to scale the estimates after the sparsification step and the subsampling step. We handle this in our algorithm by associating each subgraph with a weight, and outputting the weighted sum of the estimates of the subgraphs. The scaling factors after sparsification and subsampling are reflected by updating the weights appropriately. These weights are also used when grouping subgraphs based on the coarse estimates. Overall, the final estimate corresponds to the weighted sum of the estimates of the subgraphs. Figure 1 depicts the main components of one level of our algorithm.

We now describe the algorithms for exact counting and coarse estimation in more detail.

2.1.1 Exact Counting

Let A, B be disjoint subsets of vertices. We explain how to use BIS queries to compute $e(A, B)$, the number of edges between A and B. We use a divide and conquer approach. Let A_1, A_2 and B_1, B_2 be equipartitions of A and B, respectively. Observe that $e(A, B) = e(A_1, B_1) + e(A_1, B_2) + e(A_2, B_1) + e(A_2, B_2)$. For any pair (A_i, B_j) with no edges, we determine $e(A_i, B_j) = 0$ with one BIS query. Otherwise, we recursively determine $e(A_i, B_j)$.

We build a quadtree starting with (A, B) as the root. If $|A| = |B| = 1$, then we query this pair directly and label it with the value of $e(A, B)$, which is 0 or 1 in this case. Otherwise, we still query the pair (A, B), and if $e(A, B) = 0$, label the node as 0 and terminate. In the remaining case, we know that $e(A, B) \neq 0$, and the algorithm will recurse on the four children of (A, B), which will correspond to the pairs $(A_1, B_1), (A_1, B_2), (A_2, B_1)$, and (A_2, B_2).

To determine $e(A, B)$, the algorithm simply needs to sum the labels of all the leaves in this tree. The number of queries is equal to total number of nodes in the tree. One can prove that the number of nodes is at most $O(e(A, B) \cdot \log n) + 1$. The intuition is that the number of leaves labeled with a 1 is exactly $e(A, B)$, and the number labeled with a zero is at most $O(e(A, B) \cdot \log n) + 1$.

2.1.2 Coarse Estimator

We explain how to estimate $e(A, B)$ up to a factor of $O(\log^2 n)$ using $O(\log^3 n)$ BIS queries. As a warm-up, assume all vertices in A have degree in the range $(2^i, 2^{i+1}]$ for some $i > 0$. Sample a subset $A' \subseteq A$ by including every vertex in A independently with probability $2^i/\widetilde{e}$ for an estimate $\widetilde{e}$ we will determine later. Then, sample $B' \subseteq B$ by including every vertex in B independently with probability $1/2^i$. When $\widetilde{e} \approx e(A, B)$, there will be an edge between A' and B' with good probability, and when $\widetilde{e} \gg e(A, B) \cdot \log(n)$, then there will be no such edge with good probability. Therefore, we can perform a geometric search for $\widetilde{e} \approx e(A, B)$ to determine $e(A, B)$.

Now, we are left with the task of finding a value i^* such that a good fraction of all the edges in the graph are incident on the vertices in A that have degree roughly 2^{i^*}. By grouping vertices of degree $(2^i, 2^{i+1}]$ together, the pigeonhole principle guarantees that there is an $i^* \in \{0, \dots, \log n\}$ such that the edges touching the vertices of degree roughly 2^{i^*} is $1/\log n$ fraction of all the edges. To find the value of i^*, we perform a geometric search over the possible values $i = 1, 2, 4, 8, \dots, 2 \log n$.

The two sources of error thus come from the estimate of i^* and the acceptance probability based on the estimate for $\widetilde{e}$. Since each contributes a factor of $O(\log n)$ to the error, the coarse estimate for $e(A, B)$ will have the following guarantee.

▶ **Lemma 7.** *Let $n \geq 16$. For disjoint $A, B \subseteq [n]$, the algorithm* $\mathsf{CoarseEstimator}(A, B)$ *that makes $c_{\mathsf{ce}} \log^3 n$ BIS queries (for a constant c_{ce}) and outputs $\widetilde{e} \leq n^2$ such that with probability at least $1 - \frac{4 \log n}{n^4}$, it holds that $\frac{e(A,B)}{8 \log n} \leq \widetilde{e} \leq e(A, B) \cdot 8 \log n$.*

2.2 IS Algorithm

As with the BIS algorithm, the main building block for the IS algorithm is an efficient way to exactly count edges using IS queries. The strategy for the BIS exact counting algorithm fails because the total number of nodes in the tree can be much large than $e(A \cup B)$, up to $O(n^2)$ in the worse case. However, if we pick the sets A_1, A_2 at random, then the overall number of queries will be small with high probability. Thus, this randomized modification of the BIS exact counting algorithm computes $e(A, B)$ using $O(e(A \cup B) \log^2 n)$ IS queries with high probability. In particular, we prove the following result.

▶ **Lemma 8.** *For every disjoint $A, B \subseteq [n]$, there is a randomized algorithm that exactly computes $e(A, B)$ using at least $\frac{e(A,B)+1}{2}$ and at most $c \cdot e(A \cup B) \lceil \log^2 n \rceil + 1$ IS queries for a constant c, with probability at least $1 - \frac{1}{n^6}$.*

With this lemma in hand, we will again sparsify the graph to reduce the overall number of IS queries. In contrast to the BIS queries, we do not know how to design a coarse estimator using only IS queries. This prohibits us from designing an analogous recursive algorithm. Instead, we estimate the number of edges in one shot, by coloring the graph with a large number of colors and estimating the number of edges going between a matching of the color classes. We now discuss the reasons behind using a matching of the color classes.

An initial sparsification attempt might be to count only the edges going between a single pair of colors. If the total number of colors is $2k$, then we expect to see $m/\binom{2k}{2}$ edges between this pair. Therefore, we could set k to be large and invoke Lemma 8. Scaling by a factor of $\binom{2k}{2}$, we would hope to get an *unbiased* estimator for m.

Unfortunately, a star graph demonstrates that this approach fails, due to large variance. If we randomly color the vertices of the star graph with $2k$ colors, then out of the $\binom{2k}{2}$ pairs of color classes, only $2k-1$ pairs have any edge going across. So, if we only chose one pair of color classes, then with high probability one of the following two cases occurs: either (i) there is no edge crossing the color pair, or (ii) the number of edges crossing the pair is $\approx m/2k$. In both cases, the estimate after scaling by a factor of $\binom{2k}{2}$ is far from the truth.

At the other extreme, the vast majority of edges will be present if we look at the edges crossing *all pairs* of color classes. Indeed, the only edges we miss have both endpoints in a color class, and this accounts for only a $1/k$ fraction of the total number of edges. Thus, this does not achieve any substantial sparsification.

By using a matching of the color classes, we simultaneously get a reliable estimate of the number of edges and a sufficiently sparsified graph, as already testified by Lemma 6. Let $A_1, \cdots, A_k, B_1, \cdots, B_k$ be a random partition of the vertices into $2k$ color classes. Lemma 6 implies that the estimator $2k\sum_{i=1}^{k} e(A_i, B_i)$ is in the range $m \pm O(k\sqrt{m}\log n)$ with high probability. Hence, when k to is less than $\varepsilon\sqrt{m}/\mathrm{polylog}(n)$, we approximate m up to a factor of $(1 \pm O(\varepsilon))$. We use a geometric search to find such a k efficiently.

To bound on the number of IS queries, we claim that we can compute $\sum_{i=1}^{k} e(A_i, B_i)$ using Lemma 8, with a total of $\left(k + \frac{m}{k}\right) \cdot \mathrm{polylog}(n)$ IS queries. The first term arises since we use one query for each of the k color pairs (even if there are no edges between them). For the second term, we pay for both (i) the edges between the color classes and (ii) the total number of edges with both endpoints within a color class (since the number of IS queries in Lemma 8 scales with $e(A \cup B)$). By the sparsification lemma, we know that (i) is bounded by $O(\frac{m}{k})$ with high probability and we can prove an analogous statement for (ii). Hence, plugging in a $k \approx \frac{\varepsilon\sqrt{m}}{\mathrm{polylog}(n)}$, the total number of IS queries is bounded by $\sqrt{m} \cdot \mathrm{polylog}(n)/\varepsilon$.

2.3 Outline

The rest of the paper is organized as follows. In Section 3, we formally present the algorithm to exactly count edges between two subsets of vertices using BIS queries (Lemma 5). In Section 4, we prove our sparsification result (Lemma 10). In Section 5.1, we present the algorithm that uses BIS queries to coarsely estimate the number of edges between two subsets of vertices (Lemma 7). We combine these building blocks to construct our edge estimation algorithm using BIS queries in Section 5.2. In Section 6, we present our algorithm using IS queries. We conclude in Section 7 and mention open questions.

3 Exact Edge Counting using BIS and IS Queries

In this section, we prove Lemma 5. We give a deterministic algorithm that builds a tree by repeatedly partitioning A and B using BIS queries. The leaves of this tree identify the edges we want to count and the number of BIS queries made to construct this tree is $O(e(A,B)\log n)$. We prove Lemma 8 in the full version [2] of the paper where we give a similar randomized algorithm that uses random partitioning and makes $O(e(A \cup B)\log^2 n)$ IS queries. To present the proof, we will need the following definition.

▶ **Definition 9.** An *equipartition* of a set A with $|A| \geq 2$ is a partition A_1, A_2 of A satisfying $|A_1| = \lceil |A|/2 \rceil$ and $|A_2| = \lfloor |A|/2 \rfloor$. A *random equipartition* is an equipartition chosen uniformly at random from all equipartitions.

3.1 Proof of Lemma 5

We construct a rooted tree where every vertex is identified with a pair (A', B'), where $A' \subseteq A$ and $B' \subseteq B$, and the root of the tree is the input pair (A, B). The leaves of the tree will correspond to (A', B') with either $|A'| = |B'| = 1$ or $e(A', B') = 0$. These leaves will be labeled with 1 if $e(A', B') > 1$ and 0 otherwise. The tree has the property that every edge contributing to $e(A, B)$ appears as a leaf, and there will be exactly $e(A, B)$ leaves labeled 1. Thus, the leaves and their labels suffice to compute $e(A, B)$. Finally, this tree can be built and labeled using BIS queries.

We now proceed to describe the tree, which is constructed recursively by the following deterministic process. If $e(A, B) = 0$, then assign a 0 to the root and terminate. Otherwise, let (A', B') be the current internal node. Then

- if $e(A', B') \geq 1$, $|A'| > 1$ and $|B'| > 1$, let A'_1, A'_2 (resp. B'_1, B'_2) be an equipartition of A' (resp. B'). Add the nodes $(A'_1, B'_1), (A'_1, B'_2), (A'_2, B'_1)$ and (A'_2, B'_2) as the children.
- if $e(A', B') \geq 1$, $|A'| = 1$ and $|B'| > 1$, let B'_1, B'_2 be any equipartition of B'. Add the nodes (A', B'_1) and (A', B'_2) as the children.
- if $e(A', B') \geq 1$, $|A'| > 1$ and $|B'| = 1$, let A'_1, A'_2 be any equipartition of A'. Add the nodes (A'_1, B') and (A'_2, B') as the children.

We claim that this process uses at most $5e(A, B)\lceil \log n \rceil + 1$ BIS queries. First, note that since we make one query for each node, we simply have to bound the number of nodes. We argue that each internal node of this tree lies on a path from the root to a leaf with value 1. Indeed, if (A', B') is an internal node, then $e(A', B') \geq 1$, and hence some leaf in the sub-tree rooted at (A', B') is labeled 1.

Notice that the depth of the tree is $\lceil \log n \rceil$ and the number of leaves with value 1 is $e(A, B)$. This implies that the total number of internal nodes is at most $e(A, B)(\lceil \log n \rceil - 1)$. The number of leaves that are assigned a 0 is at most $4e(A, B)\lceil \log n \rceil + 1$ (since each node has at most 4 children). Therefore, the total number of nodes in the tree is at most $e(A, B)\lceil \log n \rceil + 4e(A, B)\lceil \log n \rceil + 1$, in turn implying that the total number of BIS queries made is at most $5e(A, B)\lceil \log n \rceil + 1$.

The number of BIS queries made is always at least $\max\{e(A, B), 1\} \geq \frac{e(A,B)+1}{2}$ since every edge with one endpoint in A and the other in B is identified.

4 Sparsification by Coloring

We present and prove our sparsification lemma. For technical reasons, we need a slightly more general sparsification statement than the one (Lemma 6) described in Section 2.

▶ **Lemma 10.** *Let $G = ([n], E)$ be a graph with m edges. For any $1 \leq k \leq \lfloor n/2 \rfloor$, let $A_1, \ldots, A_{2k}$ be a uniformly random partition of $[n]$. Then,*

(a) $\mathbb{P}\left[\left| \frac{m}{2k} - \sum_{i=1}^{k} e(A_i, A_{k+i}) \right| \geq 9\sqrt{m} \log n \right] \leq \frac{1}{n^4}$

(b) $\mathbb{P}\left[\left| \frac{m}{2k} - \sum_{i=1}^{2k} e(A_i) \right| \geq 9\sqrt{m} \log n \right] \leq \frac{1}{n^4}$.

Furthermore, for disjoint sets $A, B \subseteq [n]$ and $2 \leq k \leq \max\{|A|, |B|\}$, let $A_1, \ldots, A_k$ and $B_1, \ldots, B_k$ be uniformly random partitions of A and B, respectively. Then,

(c) $\mathbb{P}\left[\left|\frac{e(A,B)}{k} - \sum_{i=1}^{k} e(A_i, B_i)\right| \geq 9\sqrt{e(A,B)}\log n\right] \leq \frac{1}{n^4}$

Proof.

(a) Consider the random process that colors vertex t at step $t \in [n]$ with a uniformly random color $X_t \in [2k]$. The colors correspond to the partition of $[n]$ into classes $A_1, \ldots, A_{2k}$. Define $f(X_1, \ldots, X_n) = \sum_{i=1}^{k} e(A_i, A_{k+i})$, and notice that $\mathbb{E}[f] = m/(2k)$ and that $0 \leq f(X_1, \ldots, X_n) \leq m$.

When the vertex t is colored, let $N_{i,t}$ be the number of neighbors of t colored with color i among the first $t-1$ vertices. Now, define $d_t = \sum_{i \in [2k]} N_{i,t}$ to be the total number of colored neighbors of vertex t. Observe that d_t is deterministic and that $\sum_{t=1}^{n} d_t = m$ holds, since d_t is the number of edges between the vertex t and the vertices in $[t-1]$. Notice that $\mathbb{E}[N_{i,t}] = d_t/(2k)$. For $i \in [2k]$, let $\mathcal{E}_{i,t}$ be the event that

$$\left|N_{i,t} - \frac{d_t}{2k}\right| \leq 2\sqrt{d_t \ln n}. \tag{1}$$

Note that when $d_t = 0$, the event $\mathcal{E}_{i,t}$ holds with probability 1 and when $\deg(t) \geq 1$, applying Lemma 17(a) (with $r = d_t$, $\mu = \frac{d_t}{2k}$ and $s = 2\sqrt{d_t \ln n}$) gives us that $\mathcal{E}_{i,t}$ holds with probability at least $1 - 2n^{-8}$. Since $2k \leq n$, a union bound implies that with probability at least $1 - 2n^{-7}$ the events $\mathcal{E}_{1,t}, \ldots, \mathcal{E}_{2k,t}$ hold simultaneously, that is, $\mathcal{E}_t = \cap_{i \in [2k]} \mathcal{E}_{i,t}$ holds. Defining $\mathcal{E} = \cap_{t \in [n]} \mathcal{E}_t$, by a union bound we have that $\mathcal{E}$ holds with probability at least $1 - 2n^{-6}$.

To prove our claim, we will use Lemma 19, a version of Azuma's inequality that takes into account a rare bad event. We will set the bad event to be $\overline{\mathcal{E}}$. We have just argued that $\mathbb{P}[\overline{\mathcal{E}}] \leq 2n^{-6}$. Letting $c_t = 4\sqrt{d_t \ln n} + \frac{4}{n}$, we will show

$$|\mathbb{E}[f \mid X_1, \ldots, X_{t-1}, X_t = i, \mathcal{E}] - \mathbb{E}[f \mid X_1, \ldots, X_{t-1}, X_t = j, \mathcal{E}]| \leq c_t \tag{2}$$

for any two colors $i, j \in [2k]$ chosen for the vertex t. For $t \in [n]$, let M_t be the number of edges incident to the set of uncolored vertices $\{t+1, \ldots, n\}$ that go between the colored pairs $(A_1, A_{1+k}), (A_2, A_{2+k}), \ldots, (A_k, A_{2k})$. For every $i, j \in [2k]$,

$$\begin{aligned}
&|\mathbb{E}[f \mid X_1, \ldots, X_{t-1}, X_t = i, \mathcal{E}] - \mathbb{E}[f \mid X_1, \ldots, X_{t-1}, X_t = j, \mathcal{E}]| \\
&= |N_{i,t} + \mathbb{E}[M_t \mid X_1, \ldots, X_{t-1}, X_t = i, \mathcal{E}] - N_{j,t} - \mathbb{E}[M_t \mid X_1, \ldots, X_{t-1}, X_t = j, \mathcal{E}]| \\
&\leq |N_{i,t} - N_{j,t}| \\
&\quad + |\mathbb{E}[M_t \mid X_1, \ldots, X_{t-1}, X_t = i, \mathcal{E}] - \mathbb{E}[M_t \mid X_1, \ldots, X_{t-1}, X_t = j, \mathcal{E}]|,
\end{aligned}$$

where the inequality follows from the triangle inequality. Note that for any $i \in [n]$, we have $0 \leq \mathbb{E}[M_t \mid X_1, \ldots, X_{t-1}, X_t = i, \overline{\mathcal{E}}] \leq m$. Also, since M_t only involves vertices $\{t+1, \ldots, n\}$, we have, for any $j \in [n]$,

$$\mathbb{E}[M_t \mid X_1, \ldots, X_{t-1}, X_t = i] = \mathbb{E}[M_t \mid X_1, \ldots, X_{t-1}, X_t = j]. \tag{3}$$

and combining (1) and (3), and conditioning on $\mathcal{E}$,

$$\begin{aligned}
\Delta_t \;\leq\; & \max_{i,j \in [2k]} |N_{i,t} - N_{j,t}| \\
& + \frac{\mathbb{P}[\overline{\mathcal{E}}]}{\mathbb{P}[\mathcal{E}]} \left|\mathbb{E}[M_t \mid X_1, \ldots, X_t = i, \overline{\mathcal{E}}] - \mathbb{E}[M_t \mid X_1, \ldots, X_t = j, \overline{\mathcal{E}}]\right| \\
\leq\; & \max_{i,j \in [2k]} |N_{i,t} - N_{j,t}| + \frac{2m \cdot \mathbb{P}[\overline{\mathcal{E}}]}{\mathbb{P}[\mathcal{E}]} < 4\sqrt{d_t \ln n} + \frac{4}{n} = c_t.
\end{aligned}$$

To apply Lemma 19, using $\sum_{t=1}^{n} d_t = m$, we compute $\sum_t c_t^2$ as follows:

$$\begin{aligned}\sum_{t\in[n]} c_t^2 &= 16\ln n \sum_{t\in[n]} d_t + \sum_{t\in[n]} \frac{16}{n^2} + \frac{32}{n}\sum_{t\in[n]}\sqrt{d_t \ln n}\\ &= 16m\ln n + \frac{16}{n} + \frac{32}{n}\sum_{t\in[n]}\sqrt{d_t\ln n}\\ &\leq 16m\ln n + \frac{16}{n} + 32\sqrt{\frac{m\ln n}{n}} \leq 17m\ln n,\end{aligned}$$

where the penultimate inequality follows from the concavity of the square-root function. Setting $s = 9\sqrt{m}\log n - 1 > 9\sqrt{m}\ln n - 1$ and invoking Lemma 19 finishes the proof.

(b) The proof is analogous to the one in part (a) with $f(X_1,\dots,X_n) = \sum_{i=1}^{2k} e(A_i)$.

(c) Let $A = \{a_1,\dots,a_{|A|}\}$ and $B = \{b_1,\dots,b_{|B|}\}$. The vertex sequence is $a_1,\dots,a_{|A|}$ followed by $b_1,\dots,b_{|B|}$ where $X_i \in [k]$ is the color of the ith vertex, for $i \in [|A|+|B|]$. Then $f(X_1,\dots,X_n) = \sum_{i=1}^{k} e(A_i,B_i)$, and the proof is analogous to part (a). ◀

5 Edge Estimation using BIS Queries

First, we prove Lemma 7 about the coarse estimator. Then, we use this coarse estimator and importance sampling (Lemma 18), sparsification (Lemma 10), and exact edge counting (Lemma 5) to design our overall algorithm.

5.1 Coarse Estimator

We prove Lemma 7 by giving an efficient algorithm that coarsely estimates the number of edges $e(A,B)$ between A and B using $O(\log^3 n)$ BIS queries. To describe the algorithm, we will need some more notation. For a subset $S \subseteq [n]$, define $N(S)$ to be the union of the neighbors of all the vertices in S and for a vertex v, let $\deg_S(v)$ denote the number of neighbors of v that lie in S. For $i \in [\log n]$, define the set of vertices in A with degree between 2^i and 2^{i+1} as $A_i = \{v \mid v \in A,\ 2^i < \deg_B(v) \leq 2^{i+1}\}$, and let A_0 denote the vertices in A with $\deg_B(v) \leq 2$. We start with the following claim.

▶ **Claim 11.** *There exists an* $i^* \in \{0,1,\dots,\log n\}$ *such that*

$$e(A_{i^*},B) \geq \frac{e(A,B)}{\log n + 1} \qquad \text{and} \qquad |A_{i^*}| \geq \frac{e(A,B)}{2^{i^*}}\cdot\frac{1}{2(\log n + 1)}.$$

Proof. Since $\sum_{i=0}^{\log n} e(A_i,B) = e(A,B)$, the proof of the first inequality follows from averaging. To see the second inequality, observe that for every i, we have $e(A_i,B) \leq |A_i|2^{i+1}$. Hence, using the first inequality $|A_{i^*}| \geq \frac{e(A_{i^*},B)}{2^{i^*+1}} \geq \frac{e(A,B)}{2^{i^*}}\cdot\frac{1}{2(\log n+1)}$. ◀

Suppose we have an estimate $\widetilde{e}$ for $e(A,B)$. Consider CheckEstimate from Algorithm 1 for checking if $\widetilde{e}$ is correct up to logarithmic factors using logarithmically many BIS queries.

We have the following claim about the test described in Algorithm 1.

▶ **Claim 12.** *Let* $n \geq 16$. *If* $e(A,B) > 0$, *then*

(a) *if* $\widetilde{e} \geq 4e(A,B)(\log n + 1)$, CheckEstimate$((A,B),\widetilde{e})$ *accepts with probability at most* $\frac{1}{4}$.

(b) *if* $\widetilde{e} \leq \frac{e(A,B)}{4\log n}$, CheckEstimate$((A,B),\widetilde{e})$ *accepts with probability at least* $\frac{1}{2}$.

Algorithm 1: CheckEstimate$((A, B), \widetilde{e})$

```
Input: $((A, B), \widetilde{e})$ where $A, B \subseteq [n]$ are disjoint and $\widetilde{e}$ is a guess for $e(A, B)$

1 for $i = 0, 1, \dots, \log n$ do
2     Sample $A' \subseteq A$ by choosing each vertex in $A$ with probability $\min\left\{\frac{2^i}{\widetilde{e}}, 1\right\}$.
3     Sample $B' \subseteq B$ by choosing each vertex of $B$ with probability $\frac{1}{2^i}$.
4     if $e(A', B') \neq 0$ then
5         Output accept;
6     end
7 end
8 Output reject.
```

Algorithm 2: CoarseEstimator(A, B)

```
Input: $(A, B)$ where $A, B \subseteq [n]$ are disjoint
Output: An estimate $\widetilde{e}$ for the number of edges $e(A, B)$

1 if $e(A, B) = 0$ then
2     Output 0;
3 end
4 for $j = 2\log n, \dots, 0$ do
5     Run $t := 128 \log n$ independent trials of CheckEstimate$((A, B), 2^j)$.
6     if at least $\frac{3t}{8}$ of them output accept then
7         Output $2^j$;
8     end
9 end
```

Proof.

(a) For any value of the loop variable i, the probability that a fixed edge is present in the induced subgraph on A' and B' is $\min\left\{\frac{2^i}{\widetilde{e}}, 1\right\} \cdot \frac{1}{2^i} \leq \frac{1}{\widetilde{e}}$. Thus, $\mathbb{E}[e(A', B')] = \frac{e(A,B)}{\widetilde{e}} \leq \frac{1}{4(\log n+1)}$ and hence, the probability that the event $e(A', B') \neq 0$ happens for a particular value of i is $\mathbb{P}[e(A', B') \neq 0] \leq \mathbb{E}[e(A', B')] \leq \frac{1}{4(\log n+1)}$. By the union bound over the loop variable, the probability that the test accepts is at most $\frac{1}{4}$.

(b) It is enough to show that the probability is at least $\frac{1}{2}$ when the loop variable attains the value i^* where i^* is given by Claim 11. Then, we have that $|A_{i^*}| \geq \frac{e(A,B)}{2^{i^*}} \frac{1}{2(\log n+1)}$ and

$$\mathbb{P}[A' \cap A_{i^*} = \varnothing] = \left(1 - \frac{2^{i^*}}{\widetilde{e}}\right)^{|A_{i^*}|} \leq \exp\left(-\frac{m}{\widetilde{e}} \frac{1}{2(\log n + 1)}\right)$$
$$\leq \exp\left(-\frac{4 \log n}{2(\log n + 1)}\right) \leq \frac{1}{e^{1.6}},$$

where the penultimate inequality follows since $\widetilde{e} \leq \frac{e(A,B)}{4(\log n+1)}$ and the final uses $n \geq 16$. Furthermore, since $\deg_B(v) \geq 2^{i^*}$ for any $v \in A_{i^*}$, it follows that when $A' \cap A_{i^*} \neq \varnothing$, then $|N(A' \cap A_{i^*})| \geq 2^{i^*}$. So, we can bound

$$\mathbb{P}[B' \cap N(A' \cap A_{i^*}) = \varnothing \mid A' \cap A_{i^*} \neq \varnothing] \leq \left(1 - \frac{1}{2^{i^*}}\right)^{2^{i^*}} \leq \frac{1}{e}.$$

From the above, we get

$$\mathbb{P}[e(A',B') \neq 0] = \mathbb{P}[A' \cap A_{i^*} \neq \varnothing]\ \mathbb{P}[B' \cap N(A' \cap A_{i^*}) \neq \varnothing \mid A' \cap A_{i^*} \neq \varnothing]$$
$$\geq \left(1 - \frac{1}{e^{1.6}}\right)\left(1 - \frac{1}{e}\right) \geq \frac{1}{2}. \qquad \blacktriangleleft$$

Armed with the above test, we can easily estimate the number of edges up to a $O(\log n)$ factor by just doing a search, where we start with $\widetilde{e} = n^2$ and halve the number of edges each iteration. The algorithm is given in Algorithm 2 and the following claim gives an analysis.

▶ **Claim 13.** *Let $n \geq 16$.* CoarseEstimator(A, B) *outputs $\widetilde{e} \leq n^2$ satisfying $\frac{e(A,B)}{8 \log n} \leq \widetilde{e} \leq 8e(A,B)\log n$ with probability at least $1 - \frac{4 \log n}{n^4}$. The number of BIS queries made is $c_{\mathsf{ce}} \log^3 n$ for a constant c_{ce}.*

Proof. For any fixed value of j such that $2^j \geq 4(e(A,B)\log n + 1)$, the expected number of accepts is at most $\frac{t}{4}$ using Claim 12(a). The probability that we see at least $\frac{3t}{8} = \frac{t}{4} + \frac{t}{8}$ accepts can be bounded by $e^{-2t\left(\frac{1}{8}\right)^2} \leq n^{-2}$ by taking $\mu_u = \frac{t}{4}$ and $s = \frac{t}{8}$ in Lemma 17(a). By a union bound over j, the probability that none of the iterations satisfying $2^j \geq 8e(A,B)\log n \geq 4e(A,B)(\log n + 1)$ accept is at least $1 - \frac{2\log n + 1}{n^4}$ by the choice of $t = 128 \log n$.

On the other hand, when $2^j \leq \frac{e(A,B)}{4 \log n}$, the expected number of accepts is at least $\frac{t}{2}$ and so the probability that we see at least $\frac{3t}{8} = \frac{t}{2} - \frac{t}{8}$ accepts is at least $1 - e^{-2t\left(\frac{1}{8}\right)^2} \geq 1 - \frac{1}{n^4}$ by applying Lemma 17(a) with $\mu_l = \frac{t}{2}$ and $s = \frac{t}{8}$. Hence, conditioned on the event that the estimator has not accepted for any j satisfying $2^j > \frac{e(A,B)}{4 \log n}$, the probability that we accept for the unique j that satisfies $\frac{e(A,B)}{8 \log n} \leq 2^j < \frac{e(A,B)}{4 \log n}$, is at least $1 - n^{-4}$.

Overall, by the union bound, the probability of outputting an estimate $\widetilde{e}$ that does not satisfy $\frac{e(A,B)}{8 \log n} \leq \widetilde{e} \leq 8e(A,B)\log n$ is at most $\frac{2\log n + 2}{n^4} \leq \frac{4 \log n}{n^4}$. The number of queries is at most $128 \log n \cdot (2 \log n + 1)(\log n + 1) = 256 \log^3 n + O(\log^2 n)$ since for each value of j there are $t = 128 \log n$ trials of CheckEstimate, each of which makes $\log n + 1$ BIS queries. ◀

5.2 Overall BIS Algorithm (Proof of Theorem 2)

We now describe EdgeEstimator (Algorithm 3) that makes $\frac{\operatorname{polylog}(n)}{\varepsilon^4}$ BIS queries to estimate m within a factor of $(1 \pm \varepsilon)$. The subroutine BipartiteEstimator is given by Algorithm 4.

▶ **Theorem 14.** *Let $n \geq 16$. If $\frac{36 \log n}{\sqrt{m}} \leq \varepsilon \leq \frac{1}{2}$, then with probability at least $1 - \frac{\operatorname{polylog}(n)}{n^2}$*

(a) $|m - \mathsf{EdgeEstimator}(\varepsilon)| \leq \varepsilon \cdot m$*,*

(b) EdgeEstimator(ε) *uses $O\left(\frac{\log^{16} n}{\varepsilon^4}\right)$ BIS queries.*

To prove the above, we first analyze BipartiteEstimator. To this end, we need some more definitions. Let L be a list of (A, B, w), for $A, B \subseteq [n]$ that are disjoint and $w \geq 1$. For every $0 < \delta < \frac{1}{32 \log n}$, L is *good* if $|L| \leq 2t \log(n)$ and for every $(A, B, w) \in L$, $e(A, B) \geq s$, where $t = 2^{13} \cdot \frac{\log^5 n}{\delta^2}$ and $s = 81 \cdot \frac{k \log^2 n}{\delta^2}$ as defined in Step 5 of BipartiteEstimator. Define $e(L) = \sum_{(A,B,w) \in L} e(A, B)$.

We have the following guarantee on BipartiteEstimator.

▶ **Lemma 15.** *Let $n \geq 16$. If L is good and $0 < \delta < \frac{1}{32 \log n}$, then with probability at least $1 - \frac{16kt \log^2 n \cdot \log_{\lceil \frac{k}{2} \rceil} e(L)}{n^4}$, the following holds*

(a) $|\mathsf{wt}(L) - \mathsf{BipartiteEstimator}(L, k, \delta)| \leq 4\delta \log_{\lceil \frac{k}{2} \rceil} e(L) \cdot \mathsf{wt}(L)$.

Algorithm 3: EdgeEstimator(ε)

```
Input: $(n, \varepsilon)$: $n$ is the number of vertices and $\varepsilon$ is an error parameter.
Output: Estimate $\widetilde{m}$ for the number of edges $m = |E|$.

1 Partition the vertices randomly into $A$ and $B$.
2 Compute $\widetilde{e} = \mathsf{CoarseEstimator}(A, B)$.
3 if $\widetilde{e} \leq \frac{2^{20} \cdot \log^5 n}{\varepsilon^2}$ then
4     Compute $\widetilde{m} = 2 \cdot e(A, B)$ exactly using Lemma 5.
5 end
6 else
7     Compute $\widetilde{m} = 2 \cdot \mathsf{BipartiteEstimator}\left([(A, B, 1)], 4, \frac{\varepsilon}{32 \log n}\right)$ (Algorithm 4).
8 end
9 return $\widetilde{m}$.
```

Algorithm 4: BipartiteEstimator(L, k, δ)

```
Input: $(L, k, \delta)$: $L$ is a list of triples $(A, B, w)$ where $A, B \subseteq [n]$ are disjoint subsets
       and $w \geq 1$ is a positive weight for the pair $(A, B)$, $k$ is an integer, and $\delta$ is
       an error parameter.
Output: Estimate for the sum $\mathsf{wt}(L) := \sum_{(A,B,w) \in L} w \cdot e(A, B)$.

1  $L_{\mathsf{ref}} = \mathsf{Refine}(L, k)$.
2  for each $(A, B, w) \in L_{\mathsf{ref}}$ do
3      $\widetilde{e}(A, B) = \mathsf{CoarseEstimator}(A, B)$.
4  end
5  Set $t := 2^{13} \cdot \frac{\log^5 n}{\delta^2}$ and $s := 81 \cdot \frac{k \log^2 n}{\delta^2}$.
6  Define $L_{\mathsf{light}} = \{(A, B, w) \mid (A, B, w) \in L_{\mathsf{ref}},\ \widetilde{e}(A, B) \leq 8s \log n\}$.
7  Compute $\mathsf{wt}(L_{\mathsf{light}}) = \sum_{(A,B,w) \in L_{\mathsf{light}}} w \cdot e(A, B)$ exactly using Lemma 5.
8  Remove $L_{\mathsf{light}}$ from $L_{\mathsf{ref}}$.
9  For $j \in [2 \log n]$, let $S_j = \{(A, B, w) \mid (A, B, w) \in L_{\mathsf{ref}},\ w \cdot \widetilde{e}(A, B) \in (2^j, 2^{j+1}]\}$.
     Define $\widetilde{e}(S_j) = \sum_{(A,B,w) \in S_j} \widetilde{e}(A, B)$.
10 for $j = 1, \cdots, 2 \log n$ do
11     if $|S_j| > t$ then
12         Sample $t$ elements uniformly and independently from $S_j$. Denote this
             multiset by $S'_j$.
13         Update $L_{\mathsf{ref}}$ by replacing each $(A, B, w) \in S_j$ that was sampled in $S'_j$ with
             $\left(A, B, \frac{pw|S_j|}{t}\right)$, where $p$ is the number of copies of $(A, B, w)$ in $S'_j$.
14         Remove each $(A, B, w) \in S_j$ from $L_{\mathsf{ref}}$ that is not in $S'_j$.
15     end
16 end
17 Let $L_{\mathsf{sub}}$ be the current list.
18 return $\mathsf{wt}(L_{\mathsf{light}}) + \mathsf{BipartiteEstimator}(L_{\mathsf{sub}}, k, \delta)$.
```

Algorithm 5: Refine(L, k)

```
Input: $(L, k)$: $L$ is a list of triples $(A, B, w)$ where $A, B \subseteq [n]$ are disjoint subsets
       and $w \geq 1$ is a positive weight for the pair $(A, B)$, $k$ is an integer
1 for each $(A, B, w) \in L$ do
2     Partition $A$ and $B$ uniformly each into $k$ classes $A_1, \ldots, A_k$ and $B_1, \ldots, B_k$.
3     Update $L$ by replacing $(A, B, w)$ with $(A_1, B_1, wk), \cdots, (A_k, B_k, wk)$.
4 end
5 return $L$.
```

(b) *the number of BIS queries is at most* $c_{\mathsf{bp}} kst \log^4 n \cdot \log_{\lceil \frac{k}{2} \rceil} e(L)$ *where* $c_{\mathsf{bp}} = 2^{10} + 2c_{\mathsf{ce}}$ *and* c_{ce} *is the constant from Lemma 7.*

We prove Theorem 14 using Lemma 15. Lemma 15 is applied with $L = [(A, B, 1)]$, $k = 4$ and $\delta = \frac{\varepsilon}{32 \log n}$. Since $e(L) \leq n^2$, the number of BIS queries is at most $\frac{c \log^{16} n}{\varepsilon^4}$ for a constant c (after plugging in the values of k, s, t,δ).

We first informally describe BipartiteEstimator, expanding on the overview presented in Section 2. The list L, which is the input to BipartiteEstimator, corresponds to the collection of bipartite subgraphs along with their weights. The quantity $e(L)$ denotes the total number of edges in this collection without the weights. The algorithm BipartiteEstimator returns an estimate of $\mathsf{wt}(L)$, which is the weighted sum of the number of edges in the subgraphs.

At every level of the recursion, we maintain two invariants about the list we recurse on. The first is that the list size is $O(t \log n)$. This comes from the fact that we only keep a small collection of bipartite subgraphs that we recurse on. The second invariant is that for every element (A, B, w) in the list, we have $e(A, B) \geq s$. This says that we only recurse on those subgraphs that are dense. These invariants are captured in the definition of L being good. Both parameters t and s will be set to polylog(n) while k will be a constant.

Let L be the input to BipartiteEstimator. The algorithm starts with sparsifying each subgraph $(A, B, w) \in L$ by further partitioning it into k parts $(A_1, B_1), \ldots, (A_k, B_k)$. Denoting the new list by L_{ref}, Lemma 10 then guarantees that $e(L_{\mathsf{ref}}) \approx e(L)/k$ (graph is sparsified) and $\mathsf{wt}(L_{\mathsf{ref}}) \approx \mathsf{wt}(L)$ (weighted sum of the number of edges in the sparsified graph is a good estimate of the original number of edges) since we partition each $(A, B, w) \in L$ individually and increase the weight of each one by a factor of k.

Next, the algorithm BipartiteEstimator computes the coarse estimates $\widetilde{e}(A, B)$ for every (A, B, w) in L. Whenever the coarse estimate is $O(s \log n) =$ polylog(n), it computes $w \cdot e(A, B)$ exactly using the algorithm from Lemma 5. For the rest of the elements in the list which correspond to the dense subgraphs, the algorithm groups them into $2 \log n$ lists $S_1, \ldots, S_{2 \log n}$ according to the weighted coarse estimates $w \cdot \widetilde{e}(A, B)$ such that $w \cdot \widetilde{e}(A, B) \approx 2^j$ for every element in the list S_j. This allows us to use importance sampling (Lemma 18) – we can subsample t elements from each list and increase the weights by a factor of $|S_j|/t$. Since the coarse estimates are an $O(\log^2 n)$ factor approximation of the true estimates, and as we set $t =$ polylog(n), we are guaranteed that for the subsampled lists S'_j, we have that $\mathsf{wt}(S'_j) \approx \mathsf{wt}(S_j)$. The algorithm BipartiteEstimator then recurses on this subsampled collection of bipartite subgraphs and the subsampling step ensures that the size of the list we recurse on is $O(t \log n)$. Also, assuming our coarse estimates were correct, each subgraph in the subsampled list is dense, so our invariant about the input list being good is maintained.

Overall, the quantity $e(L)$ goes down by a factor of k in every level of the recursion because of sparsification, so there are $O(\log_k e(L)) = O(\log n)$ levels. Each level of the recursion incurs an additive error of $\delta/\log n$, so that the overall error is $O(\delta)$. Also, in each level of the recursion, the queries are only made by $\mathsf{CoarseEstimator}$ and for the exact counting of the edges. The dominating term comes from the exact counting and since we only count the edges exactly if $\widetilde{e}(A,B) \leq O(s\log n)$, the total number of queries made is $O(kst\log^4 n \cdot \log_k e(L)) = \text{polylog}(n)$.

We move on to prove Theorem 14 and defer the proof of Lemma 15 to the full version [2].

Proof of Theorem 14. Let A, B be the random partition chosen in Step 1 of the algorithm. Let $\mathcal{E}_1$ be the event that both

$$|2 \cdot e(A,B) - m| \leq 18\sqrt{m} \cdot \log n, \tag{4}$$

and

$$\frac{e(A,B)}{8\log n} \leq \widetilde{e} \leq e(A,B) \cdot 8\log n. \tag{5}$$

Let $\mathcal{E}_2$ be the event that when Step 7 of the algorithm is executed,

$$|2e(A,B) - \widetilde{m}| \leq \frac{\varepsilon}{2} \cdot e(A,B), \tag{6}$$

and the number of BIS queries (made in Step 7) is at most $\frac{c \cdot \log^{16} n}{\varepsilon^4}$ for a constant c to be set later. Conditioned on $\mathcal{E}_1 \cap \mathcal{E}_2$, we will bound the number of queries the algorithm makes, and we will prove that the estimate is within the desired range. Then, we will show that $\mathcal{E}_1 \cap \mathcal{E}_2$ holds with high probability.

Correctness. Observe that $\widetilde{m}$ computed in either Step 4 or Step 7 satisfies (6). Indeed, for Step 4, this follows by the setting of $\widetilde{m} = 2e(A,B)$, and for Step 7, this follows by conditioning on $\mathcal{E}_2$. Thus, the triangle inequality combined with (4) and (6) implies

$$\begin{aligned}|m - \mathsf{EdgeEstimator}(\varepsilon)| &= |m - \widetilde{m}| \leq |m - 2e(A,B)| + |2e(A,B) - \widetilde{m}| \\ &\leq 18\sqrt{m}\log n + \frac{\varepsilon}{2} \cdot e(A,B) \leq \frac{\varepsilon}{2} \cdot m + \frac{\varepsilon}{2} \cdot e(A,B) \leq \varepsilon m,\end{aligned}$$

where we used the assumption $\varepsilon \geq \frac{36\log n}{\sqrt{m}}$ and the fact $e(A,B) \leq m$.

Number of Queries. In Step 2, $\mathsf{CoarseEstimator}$ makes $O(\log^3 n)$ queries, by Lemma 7. In Step 4, the algorithm from Lemma 5 makes at most $5e(A,B) \cdot \log n + 2$ queries. This is bounded by $5e(A,B) \cdot \log n + 2 \leq 40\widetilde{e} \cdot \log^2 n + 2 = O\left(\frac{\log^7 n}{\varepsilon^2}\right)$, where we used (5) to upper bound $e(A,B)$ and the assumption on $\widetilde{e}$ in Step 3 to achieve the final bound. Finally, in Step 7, conditioned on $\mathcal{E}_2$, the number of queries is $O\left(\frac{\log^{16} n}{\varepsilon^4}\right)$.

Probability. We now analyze the probability of $\mathcal{E}_1 \cap \mathcal{E}_2$. In particular, we will show $\mathbb{P}\left[\overline{\mathcal{E}_1}\right] \leq \frac{4\log n}{n^4}$ and $\mathbb{P}\left[\overline{\mathcal{E}_2}|\mathcal{E}_1\right] \leq \frac{c'\log^8 n}{n^2}$, for a constant $c' > 0$. This suffices since then

$$\mathbb{P}\left[\overline{\mathcal{E}_1} \cup \overline{\mathcal{E}_2}\right] = \mathbb{P}\left[\overline{\mathcal{E}_1}\right] + \mathbb{P}\left[\overline{\mathcal{E}_2} \cap \mathcal{E}_1\right] \leq \mathbb{P}\left[\overline{\mathcal{E}_1}\right] + \mathbb{P}\left[\overline{\mathcal{E}_2}|\mathcal{E}_1\right] = \frac{\text{polylog}(n)}{n^2}.$$

The claim that $\mathbb{P}\left[\overline{\mathcal{E}_1}\right] \leq \frac{4\log n}{n^4}$ follows directly from Lemma 10, Lemma 7 and a union bound.

To bound $\mathbb{P}\left[\overline{\mathcal{E}_2}|\mathcal{E}_1\right]$, we invoke Lemma 15 to show that both (6) holds and the number of BIS queries is $\frac{c\log^{16} n}{\varepsilon^4}$ with high probability where $c = 4c_{\mathsf{bp}} \cdot 2^{13} \cdot 81$ and c_{bp} is the constant

Algorithm 6: EdgeEstimatorIS(n, ε)

```
Input: $(n,\varepsilon)$: $n$ is the number of vertices, and $\varepsilon$ is an error parameter.
Output: Estimate $\widetilde{m}$ for the number of edges $m = |E|$.
1 In parallel for $\widetilde{k} = 2^j$ with $1 \le 2^j \le \lfloor n/2 \rfloor$ do
2     $\widetilde{m}_{\widetilde{k}}$ = ColorCountIS$(n, \widetilde{k})$
3     Set $k^* = \left\lfloor \frac{\varepsilon \widetilde{k}}{18 \log^4 n} \right\rfloor$
4     Terminate the whole parallel for loop when the earliest iteration finishes.
5 end
6 return $\widetilde{m}$ = ColorCountIS$(n, k^*)$.
```

Algorithm 7: ColorCountIS(n, k)

```
Input: $(n,k)$: $n$ is the number of vertices, and $k$ is the number of colors.
Output: Estimate $\widetilde{m}$ for the number of edges $m = |E|$.
1 Partition the vertices into $2k$ color classes $A_1, \dots, A_{2k}$ uniformly at random.
2 Exactly compute $\widetilde{m} = \sum_{i=1}^{k} e(A_i, A_{k+i})$ using the algorithm from Lemma 8.
3 return $2k \cdot \widetilde{m}$.
```

from Lemma 15. The lemma requires that L is good, which holds because $L = [(A, B, 1)]$, $k = 4$, $\delta = \frac{\varepsilon}{32 \log n}$, and using (5),

$$e(A,B) \ge \frac{\widetilde{e}}{8 \log n} > \frac{2^{10} \cdot 81 \cdot \log^4 n}{\varepsilon^2} \ge \frac{81 k \log^2 n}{\delta^2} = s,$$

where the second inequality holds since BipartiteEstimator executes in Step 7 only when $\widetilde{e} > \frac{2^{13} \cdot 81 \cdot \log^5 n}{\varepsilon^2}$. To verify (6), we see $\mathsf{wt}([(A, B, 1)]) = e([(A, B, 1)]) = e(A, B) \le n^2$. Lemma 15 implies that with probability at least $1 - \frac{c_p \log^{10} n}{\varepsilon^2 n^4}$ (where c_p is a constant). Thus,

$$\begin{aligned}|2e(A,B) - \widetilde{m}| &= 2 \left| \mathsf{wt}([(A,B,1)]) - \mathsf{BipartiteEstimator}\left([(A,B,1)], 4, \frac{\varepsilon}{32 \log n}\right) \right| \\ &\le 8 \cdot \frac{\varepsilon}{32 \log n} \cdot \log(n^2) e(A,B) \le \frac{\varepsilon}{2} \cdot e(A,B).\end{aligned}$$

and the number of BIS queries is $O\left(\frac{\log^{16} n}{\varepsilon^4}\right)$. Since $\varepsilon \ge \frac{36 \log n}{\sqrt{m}} \ge \frac{36 \log n}{n}$, it follows that $\mathbb{P}\left[\overline{\mathcal{E}_2} | \mathcal{E}_1\right] \le \frac{c_p \log^{10} n}{\varepsilon^2 n^4} \le \frac{c' \log^8 n}{n^2}$ for a constant $c' > 0$. ◀

6 Edge Estimation using IS Queries

We briefly describe the sparsification based algorithm that uses $\sqrt{m} \cdot \mathrm{polylog}(n)/\varepsilon$ IS queries. The algorithm is given in Algorithm 6, and its guarantees are captured in Lemma 16. We defer the proof of Lemma 16 to the full version [2].

▶ **Lemma 16.** *For any $\varepsilon, m > 0$ such that $\varepsilon \ge \frac{324 \log^4 n}{\sqrt{m}}$, Algorithm 6 outputs $\widetilde{m}$ satisfying $(1-\varepsilon)m \le \widetilde{m} \le (1+\varepsilon)m$ and uses $O\left(\frac{\sqrt{m} \cdot \log^7 n}{\varepsilon}\right)$ IS queries with probability at least $1 - \frac{1}{n^3}$.*

To get Theorem 3 from the above lemma, we combine this with the folklore algorithm that computes a $(1\pm\varepsilon)$ approximation with $n^2 \cdot \text{polylog}(n)/(\varepsilon^2 m)$ edge existence queries, *i.e.*, we run both algorithms in parallel and output the estimate of the algorithm that terminates earlier. A complete analysis of the folklore algorithm can be found in the full version [2].

7 Conclusion

We studied the task of using either BIS or IS queries to estimate the number of edges in a graph. We presented randomized algorithms giving a $(1+\varepsilon)$-approximation using $\text{polylog}(n)/\varepsilon^4$ BIS queries and $\min\left\{n^2/(\varepsilon^2 m), \sqrt{m}/\varepsilon\right\} \cdot \text{polylog}(n)$ IS queries. Our algorithms estimated the number of edges by first sparsifying the graph and then exactly counting edges spanning certain bipartite subgraphs. We now describe open questions.

7.1 Open Directions

An obvious unresolved question is whether there is an algorithm to estimate the number of edges with $o(\sqrt{m})$ IS queries when $m = o(n^{4/3})$. In this context proving a lower bound of $\Omega(\sqrt{m})$ IS queries would also be very interesting. In the full version [2] of the paper we present arguments which suggest that a non-trivial lower bound might hold for IS queries.

Other open questions include using $\text{polylog}(n)$ BIS queries to estimate the number of cliques in a graph (see [9] for an algorithm using degree, neighbor and edge existence queries) or to sample a uniformly random edge (see [11] for an algorithm using degree, neighbor and edge existence queries). In general, any graph estimation problems may benefit from BIS or IS queries, possibly in combination with standard queries (such as neighbor queries). Finally, it would be interesting to know what other oracles, besides subset queries, enable estimating the number of edges (or other graph parameters) with $\text{polylog}(n)$ queries.

Acknowledgments. We thank anonymous referees for helpful comments about improving the presentation of our paper and for pointing out relevant references.

References

1 B. Aronov and S. Har-Peled. On Approximating the Depth and Related Problems. *SIAM J. Comput.*, 38(3):899–921, 2008.
2 Paul Beame, Sariel Har-Peled, Sivaramakrishnan Natarajan Ramamoorthy, Cyrus Rashtchian, and Makrand Sinha. Edge Estimation with Independent Set Oracles. *CoRR*, abs/1711.07567, 2017. `arXiv:1711.07567`.
3 Ido Ben-Eliezer, Tali Kaufman, Michael Krivelevich, and Dana Ron. Comparing the strength of query types in property testing: The case of k-colorability. *Computational Complexity*, 22(1):89–135, 2013.
4 Sergio Cabello and Miha Jejčič. Shortest paths in intersection graphs of unit disks. *Computational Geometry*, 48(4):360–367, 2015.
5 Chao L Chen and William H Swallow. Using Group Testing to Estimate a Proportion, and to Test the Binomial Model. *Biometrics*, pages 1035–1046, 1990.
6 Holger Dell and John Lapinskas. Fine-grained reductions from approximate counting to decision. *CoRR*, abs/1707.04609, 2017.
7 Robert Dorfman. The Detection of Defective Members of Large Populations. *The Annals of Mathematical Statistics*, 14(4):436–440, 1943.
8 Devdatt P Dubhashi and Alessandro Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms.* Cambridge University Press, 2009.

9 T. Eden, D. Ron, and C. Seshadhri. On Approximating the Number of k-cliques in Sublinear Time. *ArXiv e-prints*, 2017. `arXiv:1707.04858`.

10 Talya Eden, Amit Levi, Dana Ron, and C. Seshadhri. Approximately counting triangles in sublinear time. In Venkatesan Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 614–633. IEEE Computer Society, 2015.

11 Talya Eden and Will Rosenbaum. On Sampling Edges Almost Uniformly. *arXiv preprint arXiv:1706.09748*, 2017.

12 Moein Falahatgar, Ashkan Jafarpour, Alon Orlitsky, Venkatadheeraj Pichapati, and Ananda Theertha Suresh. Estimating the number of defectives with group testing. In *ISIT*, pages 1376–1380. IEEE, 2016.

13 Uriel Feige. On sums of independent random variables with unbounded variance and estimating the average degree in a graph. *SIAM Journal on Computing*, 35(4):964–984, 2006.

14 Aleksei V Fishkin. Disk graphs: A short survey. In *International Workshop on Approximation and Online Algorithms*, pages 260–264. Springer, 2003.

15 Oded Goldreich and Dana Ron. Approximating average parameters of graphs. *Random Structures & Algorithms*, 32(4):473–493, 2008.

16 Mira Gonen, Dana Ron, and Yuval Shavitt. Counting stars and other small subgraphs in sublinear-time. *SIAM J. Discrete Math*, 25(3):1365–1411, 2011.

17 Robert J Klein, Caroline Zeiss, Emily Y Chew, Jen-Yue Tsai, Richard S Sackler, Chad Haynes, Alice K Henning, John Paul SanGiovanni, Shrikant M Mane, Susan T Mayne, et al. Complement factor h polymorphism in age-related macular degeneration. *Science*, 308(5720):385–389, 2005.

18 Krzysztof Onak, Dana Ron, Michal Rosen, and Ronitt Rubinfeld. A near-optimal sublinear-time algorithm for approximating the minimum vertex cover size. *CoRR*, abs/1110.1079, 2011.

19 Dana Ron and Gilad Tsur. The power of an example: Hidden set size approximation using group queries and conditional sampling. *CoRR*, abs/1404.5568, 2014.

20 C. Seshadhri. A simpler sublinear algorithm for approximating the triangle count. *ArXiv e-prints*, 2015. `arXiv:1505.01927`.

21 Larry Stockmeyer. The complexity of approximate counting (preliminary version). In *Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing*, pages 118–126, Boston, Massachusetts, 25–27 1983.

22 Larry Stockmeyer. On approximation algorithms for #P. *SICOMP: SIAM Journal on Computing*, 14, 1985.

23 William H Swallow. Group Testing for Estimating Infection Rates and Probabilities of Disease Transmission. *Phytopathology (USA)*, 1985.

24 Jianguo Wang, Eric Lo, and Man Lung Yiu. Identifying the most connected vertices in hidden bipartite graphs using group testing. *IEEE Transactions on Knowledge and Data Engineering*, 25(10):2245–2256, 2013.

A Concentration Bounds

For proofs of the following bounds, see the book by Dubhashi and Panconesi [8].

► **Lemma 17** (Chernoff Bounds). *Let $X_1, \ldots, X_r$ be r i.i.d. random variables with $0 \le X_i \le 1$ and define $X = \sum_{i=1}^{r} X_i$. For $\mu = \mathbb{E}[X]$, let μ_l and μ_u be real numbers such that $\mu_l \le \mu \le \mu_u$.*

(a) *For any $s > 0$, we have $\mathbb{P}[X \le \mu_l - s] \le e^{-2s^2/r}$ and $\mathbb{P}[X \ge \mu_u + s] \le e^{-2s^2/r}$.*

(b) *For any $0 \le \delta < 1$, we have $\mathbb{P}[X \le (1-\delta)\mu_l] \le e^{-\frac{\mu_l \cdot \delta^2}{2}}$ and $\mathbb{P}[X \ge (1+\delta)\mu_u] \le e^{-\frac{\mu_u \cdot \delta^2}{3}}$.*

(c) *For any $\delta \ge 1$, we have $\mathbb{P}[X \ge (1+\delta)\mu_u] \le e^{-\frac{\mu_u \cdot \delta}{3}}$.*

▶ **Lemma 18** (Importance Sampling). *Let $U = \{x_1, \ldots, x_r\}$ be a set of numbers, all contained in the interval $[\frac{\alpha}{b}, \alpha b]$, for $\alpha > 0$ and $b \geq 1$. Let $\gamma > 0$ be a parameter. Consider the sum $\Gamma = \sum_{i=1}^{r} x_i$. Let X_i be a random sample chosen uniformly (and independently) from the set U, for $i = 1, \ldots, t$, and consider the estimate $Y = (r/t) \sum_{i=1}^{t} X_i$ for Γ. Then, for $t \geq \frac{b^4}{2\varepsilon^2}\left(1 + \ln \frac{1}{\gamma}\right)$, we have that $\mathbb{P}[|Y - \Gamma| \geq \varepsilon\Gamma] \leq \gamma$.*

We will need the following version of Azuma's that takes into account rare bad events.

▶ **Lemma 19.** *Let f be a function of r random variables $X_1, \ldots, X_r$ with $f(X_1, \ldots, X_r) \leq b$. Let $\mathcal{B}$ be any event and let for $i \in [r]$ let c_i satisfy*

$$\left|\mathbb{E}[f \mid X_1, \ldots, X_{i-1}, X_i = a_i, \bar{\mathcal{B}}] - \mathbb{E}[f \mid X_1, \ldots, X_{i-1}, X_i = a'_i, \bar{\mathcal{B}}]\right| \leq c_i.$$

Then for any $s > 0$, we have that

$$\mathbb{P}\left[|f - \mathbb{E}[f]| > s + b \cdot \mathbb{P}(\mathcal{B})\right] \leq \exp\left(-\frac{2s^2}{\sum_{i=1}^{r} c_i^2}\right) + \mathbb{P}[\mathcal{B}]$$