Faster Counting and Sampling Algorithms Using Colorful Decision Oracle

Anup Bhattacharya
National Institute of Science Education and Research, Bhubaneswar, India

Arijit Bishnu
Indian Statistical Institute, Kolkata, India

Arijit Ghosh
Indian Statistical Institute, Kolkata, India

Gopinath Mishra
University of Warwick, Coventry, UK

Abstract

In this work, we consider d-Hyperedge Estimation and d-Hyperedge Sample problem in a hypergraph $\mathcal{H}(U(\mathcal{H}), \mathcal{F}(\mathcal{H}))$ in the query complexity framework, where $U(\mathcal{H})$ denotes the set of vertices and $\mathcal{F}(\mathcal{H})$ denotes the set of hyperedges. The oracle access to the hypergraph is called Colorful Independence Oracle (CID), which takes d (non-empty) pairwise disjoint subsets of vertices $A_1, \ldots, A_d \subseteq U(\mathcal{H})$ as input, and answers whether there exists a hyperedge in $\mathcal{H}$ having (exactly) one vertex in each $A_i, i \in \{1, 2, \ldots, d\}$. The problem of d-Hyperedge Estimation and d-Hyperedge Sample with CID oracle access is important in its own right as a combinatorial problem. Also, Dell et al. [SODA '20] established that *decision* vs *counting* complexities of a number of combinatorial optimization problems can be abstracted out as d-Hyperedge Estimation problems with a CID oracle access.

The main technical contribution of the paper is an algorithm that estimates $m = |\mathcal{F}(\mathcal{H})|$ with $\widehat{m}$ such that

$$\frac{1}{C_d \log^{d-1} n} \leq \frac{\widehat{m}}{m} \leq C_d \log^{d-1} n.$$

by using at most $C_d \log^{d+2} n$ many CID queries, where n denotes the number of vertices in the hypergraph $\mathcal{H}$ and C_d is a constant that depends only on d. Our result coupled with the framework of Dell et al. [SODA '21] implies improved bounds for the following fundamental problems:

Edge Estimation using the Bipartite Independent Set (BIS). We improve the bound obtained by Beame et al. [ITCS '18, TALG '20].

Triangle Estimation using the Tripartite Independent Set (TIS). The previous best bound for the case of graphs with low *co-degree* (Co-degree for an edge in the graph is the number of triangles incident to that edge in the graph) was due to Bhattacharya et al. [ISAAC '19, TOCS '21], and Dell et al.'s result gives the best bound for the case of general graphs [SODA '21]. We improve both of these bounds.

Hyperedge Estimation & Sampling using Colorful Independence Oracle (CID). We give an improvement over the bounds obtained by Dell et al. [SODA '21].

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms

Keywords and phrases Query Complexity, Subset Query, Hyperedge Estimation, and Colorful Independent Set oracle

Digital Object Identifier 10.4230/LIPIcs.STACS.2022.10

Related Version *Full Version*: https://arxiv.org/abs/2201.04975

1 Introduction

Estimating different combinatorial structures like edges, triangles and cliques in an unknown graph that can be accessed only through *query oracles* is a fundamental area of research in *sublinear algorithms* [13, 14, 11, 12]. Different query oracles provide unique ways of looking at the same graph. Beame et al. [1] introduced an independent set based subset query oracle, named Bipartite Independent Set (BIS) query, to estimate the number of edges in a graph using polylogarithmic queries. The BIS query answers a YES/NO question on the existence of an edge between two disjoint subsets of vertices of a graph G. The next natural questions in this line of research were problems of estimation and uniform sampling of hyperedges in hypergraphs [9, 3, 4]. In this paper, we will be focusing on these two fundamental questions, and in doing so, we will improve all the previous results [2, 9, 3, 4].

1.1 Our query oracle, results and the context

A hypergraph $\mathcal{H}$ is a *set system* $(U(\mathcal{H}), \mathcal{F}(\mathcal{H}))$, where $U(\mathcal{H})$ denotes a set of n vertices and $\mathcal{F}(\mathcal{H})$, a set of subsets of $U(\mathcal{H})$, denotes the set of hyperedges. A hypergraph $\mathcal{H}$ is said to be *d-uniform* if every hyperedge in $\mathcal{H}$ consists of exactly d vertices. The cardinality of the hyperedge set is denoted as $m(\mathcal{H}) = |\mathcal{F}(\mathcal{H})|$. We will access the hypergraph using the following oracle[1] [6].

▶ **Definition 1.1** (Colorful Independent Set (CID)). *Given d pairwise disjoint subsets of vertices $A_1, \ldots, A_d \subseteq U(\mathcal{H})$ of a hypergraph $\mathcal{H}$ as input,* CID *query answers* Yes *if and only if $m(A_1, \ldots, A_d) \neq 0$, where $m(A_1, \ldots, A_d)$ denotes the number of hyperedges in $\mathcal{H}$ having exactly one vertex in each A_i, where $i \in \{1, 2, \ldots, d\}$.*

Note that the earlier mentioned BIS is a special case of CID when $d = 2$. With this query oracle access, we solve the following two problems.

d-Hyperedge-Estimation
Input: Vertex set $U(\mathcal{H})$ of a hypergraph $\mathcal{H}$ with n vertices, a CID oracle access to $\mathcal{H}$, and $\varepsilon \in (0, 1)$.
Output: A $(1 \pm \varepsilon)$-approximation $\widehat{m}$ to $m(\mathcal{H})$ with probability $1 - 1/n^{\Omega(d)}$.

Note that Edge Estimation problem is a special case of d-Hyperedge-Estimation when $d = 2$.

d-Hyperedge-Sample
Input: Vertex set $U(\mathcal{H})$ of a hypergraph $\mathcal{H}$ with n vertices, a CID oracle access to $\mathcal{H}$, and $\varepsilon \in (0, 1)$.
Output: With probability $1 - 1/n^{\Omega(d)}$, report a sample from a distribution of hyperedges in $\mathcal{H}$ such that the probability that any particular hyperedge is sampled lies in the interval $\left[(1-\varepsilon)\frac{1}{m}, (1+\varepsilon)\frac{1}{m}\right]$.

This area started with the investigation of Edge Estimation problem by Dell and Lapinskas [7, 8] and Beame et al. [1], then Bhattacharya et al. [3, 4] studied d-Hyperedge-Estimation for $d = 3$, and more recently Dell et al. [9] gave algorithms for d-Hyperedge-Estimation and d-Hyperedge-Sample for general d. Beame et al. [1] showed that Edge

[1] In [6], the oracle is named as Generalized Partite Independent Set oracle. Here, we follow the same suit as Dell et al. [9] with respect to the name of the oracle.

Estimation problem can be solved using $\mathcal{O}\left(\frac{\log^{14} n}{\varepsilon^4}\right)$ BIS queries. Having estimated the number of edges in a graph using BIS queries, a very natural question was to estimate the number of hyperedges in a hypergraph using an appropriate query oracle. This extension is nontrivial as two edges in a graph can intersect in at most one vertex but the intersection pattern between two hyperedges in a hypergraph is more complicated. As a first step towards resolving this question, Bhattacharya et al. [3, 4] considered d-Hyperedge-Estimation in 3-uniform hypergraphs using CID queries. They showed that when *co-degree* of any pair of vertices in a 3-uniform hypergraph is bounded above by Δ, then one can solve d-Hyperedge-Estimation using $\mathcal{O}\left(\frac{\Delta^2 \log^{18} n}{\varepsilon^4}\right)$ CID queries. Recall that co-degree of two vertices in a hypergraph is the number of hyperedges that contain both vertices. Dell et al. [9] generalized the results of Beame et al. [1] and Bhattacharya et al. [3, 4], and obtained a similar (with an improved dependency in terms of ε) result for the d-Hyperedge-Estimation problem for general d. Apart from d-Hyperedge-Estimation problem, they also considered the problem of d-Hyperedge-Sample. The results of Dell et al. [9] are formally stated in the following proposition:

▶ **Proposition 1.2** (Dell et al. [9]). *d-Hyperedge-Estimation and d-Hyperedge-Sample can be solved by using $\mathcal{O}_d\left(\frac{\log^{4d+8} n}{\varepsilon^2}\right)$ and $\mathcal{O}_d\left(\frac{\log^{4d+12} n}{\varepsilon^2}\right)$ CID queries, respectively.* [2]

Currently, the best known bound (prior to this work) for solving d-Hyperedge-Estimation problem, for general d, is due to Dell et al. [9], but note that for constant $\varepsilon \in (0,1)$, Beame et al. [1, 2] still have the best bound for the Edge Estimation problem.

Our main result is an improved *coarse estimation* technique, named Rough Estimation, and is stated in the following theorem. The significance of the coarse estimation technique will be discussed in Section 1.2.

▶ **Theorem 1.3** (Main result). *There exists an algorithm* Rough Estimation *that has* CID *query access to a d-uniform hypergraph $\mathcal{H}(U,\mathcal{F})$ and returns $\widehat{m}$ as an estimate for $m = |\mathcal{F}(\mathcal{H})|$ such that*

$$\frac{1}{C_d \log^{d-1} n} \leq \frac{\widehat{m}}{m} \leq C_d \log^{d-1} n$$

with probability at least $1 - 1/n^{\Omega(d)}$ using at most $C_d \log^{d+2} n$ CID queries, where C_d is a constant that depends only on d and n denotes the number of vertices in $\mathcal{H}$.

Coarse estimation gives a crude polylogarithmic approximation for m, the number of hyperedges in $\mathcal{H}$. This improvement in the coarse estimation algorithm coupled with *importance sampling* and the algorithmic framework of Dell et al. [9] gives an improved algorithm for both d-Hyperedge-Estimation and d-Hyperedge-Sample problems.

▶ **Theorem 1.4** (Improved bounds for estimating and sampling). *d-Hyperedge-Estimation and d-Hyperedge-Sample problems can be solved by using $\mathcal{O}_d\left(\frac{\log^{3d+5} n}{\varepsilon^2}\right)$ and $\mathcal{O}_d\left(\frac{\log^{3d+9} n}{\varepsilon^2}\right)$ CID queries, respectively.*

---

[2] Dell et al. [9] studied d-Hyperedge-Estimation and d-Hyperedge-Sample where the probability of success is $1-\delta$ for some given $\delta \in (0,1)$, and have showed that d-Hyperedge-Estimation and d-Hyperedge-Sample can be solved by using $\mathcal{O}_d\left(\frac{\log^{4d+7} n}{\varepsilon^2} \log \frac{1}{\delta}\right)$ and $\mathcal{O}_d\left(\frac{\log^{4d+11} n}{\varepsilon^2} \log \frac{1}{\delta}\right)$ CID queries, respectively. In Proposition 1.2, we have taken $\delta = n^{\mathcal{O}(d)}$. But both the results of Beame et al. [1, 2] and Bhattacharya et al. [3, 4] are in the high probability regime.
In this paper, we work with success probability to be $1 - 1/n^{\Omega(d)}$ for simplicity of presentation and compare our results with all previous results in a high probability regime.

The details regarding how Theorem 1.3 can be used together with the framework of Dell et al. [9] to prove Theorem 1.4 will be discussed in Section 5.

Using Theorem 1.4, we directly get the following improved bounds for Edge Estimation and d-Hyperedge-Estimationin 3-uniform hypergraph by substituting $d = 2$ and $d = 3$, respectively.

▶ **Corollary 1.5.**

(a) Edge Estimation *can be solved using* $\mathcal{O}\left(\frac{\log^{11} n}{\varepsilon^2}\right)$ *queries to* Bipartite Independent Set *(*BIS*) oracle.*

(b) d-Hyperedge-Estimation *in a* 3*-uniform hypergraph can be solved using* $\mathcal{O}\left(\frac{\log^{14} n}{\varepsilon^2}\right)$ CID *queries.*

The above corollary gives the best bound (till now) for the Edge Estimation. Recall that Bhattacharya et al. [3, 4] proved that when the co-degree of a 3-uniform graph is bounded by Δ then d-Hyperedge-Estimation in that hypergraph can be solved using $\mathcal{O}\left(\frac{\Delta^2 \log^{18} n}{\varepsilon^4}\right)$ CID queries. For fixed $\varepsilon \in (0, 1)$ and $\Delta = o(\log n)$ the bound obtained by Bhattacharya et al. [3, 4] is asymptotically better than the bound we get from Dell et al. [9], see Proposition 1.2. Note that Corollary 1.5 (b) improves the bounds obtained by Bhattacharya et al. [3, 4] and Dell et al. [9] for all values of Δ and $\varepsilon \in (0, 1)$.

1.2 Fundamental role of coarse estimation

The framework of Dell et al. [9] is inspired by the following observation. Let us consider $t = \mathcal{O}\left(\frac{\log n}{\varepsilon^2}\right)$ independent subhypergraphs each induced by $n/2$ uniform random vertices. The probability, that a particular hyperedge is present in a subhypergraph induced by $n/2$ many uniform random vertices, is $\frac{1}{2^d}$. Denoting X as the sum of the numbers of the hyperedges present in the t subhypergraphs, observe that $\frac{2^d}{t}X$ is a $(1 \pm \varepsilon)$-approximation of m. If we repeat the procedure recursively $\mathcal{O}(\log n)$ times, then all the subhypergraphs will have a bounded number of vertices in terms of d, at which point the number of hyperedges can be determined exactly by using $\mathcal{O}_d(1)$ CID queries. However, the number of induced subhypergraphs in the worst case can become as *large* as $\Omega\left((\log n)^{\log n}\right)$.

To have the number of subhypergraphs bounded at all point of time, they use *importance sampling*. It is about maintaining the weighted sum of some variables whose approximate value is known to us. The output will be a bounded number of variables and some weight parameters such that the weighted sum of the variables estimates the required sum. The objective of the importance sampling procedure in Beame et al. [1, 2] and Bhattacharya et al. [3, 4], are also the same [3]. However, Dell et al. improved the importance sampling result by the use of a particular form of Bernstein inequality and by a very careful analysis.

To apply importance sampling, it is required to have a rough estimate (possibly with a polylogarithmic approximation factor) of the number of hyperedges in each subhypergraph that are currently present for processing – this is what exactly coarse estimation does. The objective of coarse estimation in Beame et al. [1, 2] and Bhattacharya et al. [3, 4] are also the same [4]. But all these frameworks have a commonality. The approximation guarantee and the query complexity of the coarse estimation has a direct bearing on the query complexity of the final algorithm.

[3] In fact, Bhattacharya et al. [3, 4] directly use the importance sampling developed by Beame et al. [1, 2]

[4] Note that the main merit of the framework of Dell et al. [9] over Beame et al. [1, 2] and Bhattacharya et al. [3, 4] is not only that it generalized to hypergraph, but also the dependence on ε is $1/\varepsilon^2$ in Dell et al. [9]'s work as opposed to $\frac{1}{\varepsilon^4}$ in Beame et al. [1, 2] and Bhattacharya et al. [3, 4].

Therefore, any improvement in the coarse estimation algorithm will directly improve the query complexities of d-Hyperedge-Estimation and d-Hyperedge-Sample. In this paper, we focus on improving the coarse estimation algorithm.

1.3 Setup and notations

We denote the sets $\{1,\ldots,n\}$ and $\{0,\ldots,n\}$ by $[n]$ and $[n^*]$, respectively. A hypergraph $\mathcal{H}$ is a *set system* $(U(\mathcal{H}),\mathcal{F}(\mathcal{H}))$, where $U(\mathcal{H})$ denotes the set of vertices and $\mathcal{F}(\mathcal{H})$ denotes the set of hyperedges. The set of vertices present in a hyperedge $F \in \mathcal{F}(\mathcal{H})$ is denoted by $U(F)$ or simply F. A hypergraph $\mathcal{H}$ is said to be d-uniform if all the hyperedges in $\mathcal{H}$ consist of exactly d vertices. The cardinality of the hyperedge set is $m(\mathcal{H}) = |\mathcal{F}(\mathcal{H})|$. For $A_1,\ldots,A_d \subseteq U(\mathcal{H})$ (not necessarily pairwise disjoint), $\mathcal{F}(A_1,\ldots,A_d) \subseteq \mathcal{F}(\mathcal{H})$ denotes the set of hyperedges having a vertex in each A_i, and $m(A_1,\ldots,A_d)$ is the number of hyperedges in $|\mathcal{F}(A_1,\ldots,A_d)|$.

Let $\mathbb{E}[X]$ and $\mathbb{V}[X]$ denote the expectation and variance of the random variable X. For an event $\mathcal{E}$, the complement of $\mathcal{E}$ is denoted by $\overline{\mathcal{E}}$. The statement "a is a $(1\pm\varepsilon)$-approximation of b" means $|b-a| \leq \varepsilon \cdot b$. For $x \in \mathbb{R}$, $\exp(x)$ denotes the standard exponential function e^x. In this paper, d is a constant, and $\mathcal{O}_d(\cdot)$ and $\Omega_d(\cdot)$ denote the standard $\mathcal{O}(\cdot)$ and $\Omega(\cdot)$, where the constant depends only on d. We use $\log^k n$ to denote $(\log n)^k$. By polylogarithmic, we mean $\mathcal{O}_d\left(\frac{\log^{\mathcal{O}(d)} n}{\varepsilon^{\Omega(1)}}\right)$ in this paper.

1.4 Paper organization

In Section 2, we describe the notion of an *ordered hyperedge*, and define three other query oracles that can be simulated by using $\mathcal{O}_d(\log n)$ CID queries. The role of ordered hyperedges and these oracles are mostly expository purposes, i.e., they help us to describe our algorithms and the calculations more neatly. Section 3 gives a brief overview of the proof of our main technical result. In Section 4 we give the proof of our main result (Theorem 1.3). We describe in Section 5 implications of our main result and how Theorem 1.3 can be used to prove Theorem 1.4. The equivalence proofs of the CID oracle and its variants are discussed in Section 2. Some useful probability results are given in Appendix A. Since we use different types of oracles in the calculations, we have recalled all their definitions in Appendix B for the ease of reference. Proofs omitted are marked with $\star$, and can be found in the full version [5] of this paper.

2 Preliminaries: Ordered hyperedges, CID oracle, and its variants

Ordered hyperedges

We will use the subscript "o" to denote the set of ordered hyperedges. For example, $\mathcal{H}_o(U,\mathcal{F}_o)$ denotes the ordered hypergraph corresponding to $\mathcal{H}(U,\mathcal{F})$. Here $\mathcal{F}_o(\mathcal{H})$ denotes the set of ordered hyperedges that contains $d!$ ordered d-tuples for each hyperedge in $\mathcal{H}(U,\mathcal{F})$. Let $m_o(\mathcal{H}_o)$ denotes $|\mathcal{F}_o(\mathcal{H}_o)|$. Note that $m_o(\mathcal{H}_o) = d!m(\mathcal{H})$. Also, let $\mathcal{F}_o(A_1,\ldots,A_d)$ denotes the set $\{F_o \in \mathcal{F}_o(\mathcal{H}) : \text{the } i\text{-th vertex of } F_o \text{ is in } A_i, \forall i \in [d]\}$. The corresponding number for ordered hyperedges is $m_o(A_1,\ldots,A_d)$. Note that $\mathcal{F}_o(U(\mathcal{H}),\ldots,U(\mathcal{H})) = \mathcal{F}_o(\mathcal{H})$.

CID oracle and its variants

Note that the CID query takes as input d pairwise disjoint subsets of vertices. We now define two related query oracles CID_1 and CID_2 that remove the disjointness requirements for the input. Then we extent CID_2 to the ordered setting. We show that both query oracles can be

simulated, with high probability, by making $\mathcal{O}_d(\log n)$ queries to the CID oracle. The oracles CID_1 and CID_2 will be used in the description of the algorithm for ease of exposition.

CID_1: Given s pairwise disjoint subsets of vertices $A_1, \ldots, A_s \subseteq U(\mathcal{H})$ of a hypergraph $\mathcal{H}$ and $a_1, \ldots, a_s \in [d]$ such that $\sum_{i=1}^{s} a_i = d$, CID_1 query on input $A_1^{[a_1]}, A_2^{[a_2]}, \cdots, A_s^{[a_s]}$ answers YES if and only if $m(A_1^{[a_1]}, \ldots, A_s^{[a_s]}) \neq 0$. Here $A^{[a]}$ denotes the set A repeated a times.

CID_2: Given any d subsets of vertices $A_1, \ldots, A_d \subseteq U(\mathcal{H})$ of a hypergraph $\mathcal{H}$, CID_2 query on input $A_1, \ldots, A_d$ answers YES if and only if $m(A_1, \ldots, A_d) \neq 0$.

CID_2^o: Given any d subsets of vertices $A_1, \ldots, A_d \subseteq U(\mathcal{H}_o)$ of an ordered hypergraph $\mathcal{H}_o$, CID_2^o query on input $A_1, \ldots, A_d$ answers YES if and only if $m_o(A_1, \ldots, A_d) \neq 0$.

Observe that the CID_2 query is the same as the CID query without the requirement that the input sets are disjoint. For the CID_1 query, multiple repetitions of the same set is allowed in the input. It is obvious that a CID query can be simulated by a CID_1 or CID_2 query. Also, CID_2^o is the ordered analogue of CID_2 . Using the following observation, we show how a CID_2^o, CID_1 , or a CID_2 query can be simulated by a polylogarithmic number of CID queries.

▶ **Observation 2.1** (⋆, Connection between query oracles). *Let $\mathcal{H}(U, \mathcal{F})$ denote a hypergraph and $\mathcal{H}_o(U, \mathcal{F}_o)$ denote the corresponding ordered hypergraph.*

(i) *A CID_1 query to $\mathcal{H}(U, \mathcal{F})$ can be simulated using $\mathcal{O}_d(\log n)$ CID queries with probability $1 - 1/n^{\Omega(d)}$.*

(ii) *A CID_2 query $\mathcal{H}(U, \mathcal{F})$ can be simulated using $\mathcal{O}_d(1)$ CID_1 queries.*

(iii) *A CID_2 query $\mathcal{H}(U, \mathcal{F})$ can be simulated using $\mathcal{O}_d(\log n)$ CID queries with probability $1 - 1/n^{\Omega(d)}$.*

(iv) *A CID_2^o query to $\mathcal{H}_o(U, \mathcal{F}_o)$ can be simulated using a CID_2 query to $\mathcal{H}(U, \mathcal{F})$.*

3 Overview of the main structural result

To prove Theorem 1.3, we first consider Lemma 3.1, which is the central result of the paper and is the ordered hypergraph analogue of Theorem 1.3. The main theorem (Theorem 1.3) follows from Lemma 3.1 along with Observation 2.1.

▶ **Lemma 3.1** (Main Lemma). *There exists an algorithm* ROUGH ESTIMATION *that has CID_2^o query access to a d-uniform ordered hypergraph $\mathcal{H}_o(U, \mathcal{F}_o)$ corresponding to hypergraph $\mathcal{H}(U, \mathcal{F})$ and returns $\widehat{m}_o$ as an estimate for $m_o = |\mathcal{F}_o(\mathcal{H}_o)|$ such that*

$$\frac{1}{C_d \log^{d-1} n} \leq \frac{\widehat{m}}{m} \leq C_d \log^{d-1} n$$

with probability at least $1 - 1/n^{\Omega(d)}$ using at most $C_d \log^{d+1} n$ CID_2^o queries, where C_d is a constant that depends only on d.

At a high level, the idea for an improved coarse estimation involves a recursive bucketing technique and careful analysis of the intersection pattern of hypergraphs.

To build up towards the final proof, we need to prove Lemma 3.1. Towards this end, we first define some quantities and prove Claim 3.2. For that, let us think of partitioning the vertex set in $U_1 = U(\mathcal{H})$ into buckets such that the vertices in each bucket appear as the first vertex in approximately the same number of hyperedges. So, there will be at most $d \log n + 1$ buckets. It can be shown that that there is a bucket $Z_1 \subseteq U_1$ such that the number of hyperedges, having the vertices in the bucket as the first vertex, is at least $\frac{m_o}{d \log n + 1}$. For each vertex $z_1 \in Z_1$, let the number of hyperedges in $\mathcal{H}_o$, having z_1 as the first vertex, lie between 2^{q_1} and $2^{q_1+1} - 1$ for some suitable q_1. Then we can argue that

$$|Z_1| \geq \frac{m_o}{2^{q_1+1}(d\log n+1)}.$$

Similarly, we extend the bucketing idea to tuples as follows. Consider a vertex a_1 in a particular bucket of U_1 and consider all the ordered hyperedges in $\mathcal{F}_o(a_1)$ containing a_1 as the first vertex. We can bucket the vertices in $U_2 = U(\mathcal{H})$ such that the vertices in each bucket of U_2 are present in approximately the same number of hyperedges in $\mathcal{F}_o(a_1)$ as the second vertex. We generalize the above bucketing strategy with the vertices in U_i's, which is formally described below. Notice that this way of bucketing will allow us to use conditionals on sampling vertices from the desired buckets of U_i's.

For $q_1 \in [(d\log n)^*]$, let $U_1(q_1) \subseteq U_1$ be the set of vertices in $a_1 \in U_1$ such that for each $a_1 \in U_1(q_1)$, the number of hyperedges in $\mathcal{F}_o(\mathcal{H}_o)$, containing a_1 as the first vertex, lies between 2^{q_1} and $2^{q_1+1}-1$. For $2 \leq i \leq d-1$, and $q_j \in [(d\log n)^*]$ for each $j \in [i-1]$, consider $a_1 \in U_1(q_1), a_2 \in U_2((q_1,a_1),q_2),\ldots,\ a_{i-1} \in U_{i-1}((q_1,a_1),\ldots,(q_{i-2},a_{i-2}),q_{i-1})$. Let $U_i((q_1,a_1),\ldots,(q_{i-1},a_{i-1}),q_i)$ be the set of vertices in U_i such that for each $u_i \in U_i((q_1,u_1),\ldots,(q_{i-1},a_{i-1}),q_i)$, the number of ordered hyperedges in $\mathcal{F}_o(\mathcal{H}_o)$, containing u_j as the j-th vertex for all $j \in [i]$, lies between 2^{q_i} and $2^{q_i+1}-1$. We need the following result to proceed further. For ease of presentation, we use (Q_i, A_i) to denote $(q_1,a_1),\ldots,(q_{i-1},a_{i-1})$ for $2 \leq i \leq d-1$. Informally, Claim 3.2 says that for each $i \in [d-1]$, there exists a bucket in U_i having a *large* number of vertices contributing approximately the same number of hyperedges..

▷ Claim 3.2 (⋆).

(i) There exists $q_1 \in [(d\log n)^*]$ such that

$$|U_1(q_1)| > \frac{m_o(\mathcal{H}_o)}{2^{q_1+1}(d\log n+1)}.$$

(ii) Let $2 \leq i \leq d-1$ and $q_j \in [(d\log n)^*]\ \forall j \in [i-1]$. Let $a_1 \in U_1(q_1)$, $a_j \in U_j((Q_{j-1},A_{j-1}),q_j)\ \forall j \neq 1$ and $j < i$. There exists $q_i \in [(d\log n)^*]$ such that

$$|U_i((Q_i,A_i),q_i)| > \frac{2^{q_{i-1}}}{2^{q_i+1}(d\log n+1)}.$$

4 Proof of Lemma 3.1

We now prove Lemma 3.1 formally. The algorithm corresponding to Lemma 3.1 is Algorithm 2 (named Rough Estimation). Algorithm 1 (named Verify-Estimate) is a subroutine of Algorithm 2. Algorithm 1 determines whether a given estimate $\widehat{R}$ of the number of ordered hyperedges is correct up to $\mathcal{O}_d(\log^{2d-3} n)$ factor. Lemma 4.1 and 4.2 are intermediate results needed to prove Lemma 3.1; they bound the probability from above and below, respectively of Verify-Estimate accepting the estimate $\widehat{\mathcal{R}}$.

▶ **Lemma 4.1.** *If* $\widehat{\mathcal{R}} \geq 20d^{2d-3}4^d\ m_o(\mathcal{H}_o)\log^{2d-3} n$, *then*

$$\mathbb{P}(\text{Verify-Estimate } (\mathcal{H}_o, \widehat{\mathcal{R}}) \text{ accepts the estimate } \widehat{R}) \leq \frac{1}{20\cdot 2^d}.$$

Algorithm 1 VERIFY-ESTIMATE $(\mathcal{H}_o, \widehat{\mathcal{R}})$.

Input: CID query access to a d-uniform hypergraph $\mathcal{H}_o(U, \mathcal{F})$ and a guess $\widehat{R}$ for the number of hyperedges in $\mathcal{H}_o$.
Output: ACCEPT $\widehat{\mathcal{R}}$ or REJECT $\widehat{\mathcal{R}}$.

```
Let
    U_1 = ... = U_d = U(H) for (j_1 = d log n to 0) do
        find B_1 ⊆ U_1 by sampling every element of U_1 with probability p_1 = min{2^{j_1}/R̂, 1}
         independently of other elements.
        for (j_2 = d log n to 0) do
            find B_2 ⊆ U_2 by sampling every element of U_2 with probability
             p_2 = min{2^{j_2 - j_1} · d log n, 1} independently of other elements.
            ⋮
            for (j_{d-1} = d log n to 0) do
                find B_{d-1} ⊆ U_{d-1} by sampling every element of U_{d-1} with probability
                 p_{d-1} = min{2^{j_{d-1} - j_{d-2}} · d log n, 1} independently of other elements.
                Let j = (j_1, ..., j_{d-1}) ∈ [(d log n)*]^{d-1}
                Let p(i, j) = p_i, where 1 ≤ i ≤ d − 1
                Let B(i, j) = B_i, where 1 ≤ i ≤ d − 1
                find B(d, j) = B_d ⊆ U_d by sampling every element of U_d with probability
                 p_d = min{2^{-j_{d-1}}, 1} independently of other elements.
                if (m_o(B_{1,j}, ..., B_{d,j}) ≠ 0) then
                    ACCEPT /*[Note that CID_2^o query is called in the above line.]*/
                end
            end
        end
    end
    REJECT
```

Proof. Consider the set of ordered hyperedges $\mathcal{F}_o(\mathcal{H}_o)$ in $\mathcal{H}_o$. Algorithm VERIFY-ESTIMATE taking parameters $\mathcal{H}_o$, and $\widehat{\mathcal{R}}$ and described in Algorithm 1, loops over all possible $\mathbf{j} = (j_1, \ldots, j_{d-1}) \in [(d \log n)^*]^{d-1}$ [5]. For each $\mathbf{j} = (j_1, \ldots, j_{d-1}) \in [(d \log n)^*]^{d-1}$, VERIFY-ESTIMATE $(\mathcal{H}_o, \widehat{\mathcal{R}})$ samples vertices in each U_i with *suitable* probability values $p(i, \mathbf{j})$, depending on $\mathbf{j}$, $\widehat{R}$, d and $\log n$, to generate the sets $B_{i,\mathbf{j}}$ for $1 \leq i \leq d$. See Algorithm 1 for the exact values of $p(i, \mathbf{j})$'s. VERIFY-ESTIMATE $(\mathcal{H}_o, \widehat{\mathcal{R}})$ reports ACCEPT if there exists one $\mathbf{j} \in [(d \log n)^*]^{d-1}$ such that $m_o(B_{1,\mathbf{j}}, \ldots, B_{d,\mathbf{j}}) \neq 0$. Otherwise, REJECT is reported by VERIFY-ESTIMATE $(\mathcal{H}_o, \widehat{\mathcal{R}})$.

For an ordered hyperedge $F_o \in \mathcal{F}_o(\mathcal{H}_o) = \mathcal{F}_o(U_1, \ldots, U_d)$ and $\mathbf{j} \in [(d \log n)^*]^{d-1}$. Note that

$$U_1 = \ldots = U_d = U(\mathcal{H}).$$

Let $X^{\mathbf{j}}_{F_o}$ denote the indicator random variable such that $X^{\mathbf{j}}_{F_o} = 1$ if and only if $F_o \in \mathcal{F}_o(B_{1,\mathbf{j}}, \ldots, B_{d,\mathbf{j}})$. Let

$$X_{\mathbf{j}} = \sum_{F_o \in \mathcal{F}_o(\mathcal{H}_o)} X^{\mathbf{j}}_{F_o}.$$

[5] Recall that $[n]^*$ denotes the set $\{0, \ldots, n\}$.

Note that $m_o(B_{1,\mathbf{j}}, \ldots, B_{d,\mathbf{j}}) = X_{\mathbf{j}}$. We have,

$$\begin{aligned}\mathbb{P}\left(X^{\mathbf{j}}_{F_o} = 1\right) &= \prod_{i=1}^{d}(p(i,\mathbf{j})) \\ &\leq \frac{2^{j_1}}{\widehat{\mathcal{R}}} \cdot \frac{2^{j_2}}{2^{j_1}} d\log n \times \cdots \times \frac{2^{j_{d-1}}}{2^{j_{d-2}}} d\log n \times \frac{1}{2^{j_{d-1}}} \\ &= \frac{d^{d-2}\log^{d-2} n}{\widehat{\mathcal{R}}}\end{aligned}$$

Then,

$$\mathbb{E}\left[X_{\mathbf{j}}\right] \leq \frac{m_o(\mathcal{H}_o)}{\widehat{\mathcal{R}}} d^{d-2}\log^{d-2} n,$$

and since $X_{\mathbf{j}} \geq 0$, we have

$$\mathbb{P}\left(X_{\mathbf{j}} \neq 0\right) = \mathbb{P}(X_{\mathbf{j}} \geq 1) \leq \mathbb{E}\left[X_{\mathbf{j}}\right] \leq \frac{m_o(\mathcal{H}_o)}{\widehat{\mathcal{R}}} d^{d-2}\log^{d-2} n.$$

Now, using the fact that $\widehat{\mathcal{R}} \geq 20d^{2d-3} \cdot 4^d \cdot m_o(\mathcal{H}_o)\log^{2d-3} n$, we have

$$\mathbb{P}\left(X_{\mathbf{j}} \neq 0\right) \leq \frac{1}{20d^{d-1} \cdot 4^d \cdot \log^{d-1} n}.$$

Recall that Verify-Estimate accepts if and only if there exists $\mathbf{j}$ such that $X_{\mathbf{j}} \neq 0$ [6]. Using the union bound, we get

$$\begin{aligned}\mathbb{P}\left(\text{Verify-Estimate } (\mathcal{H}_o, \widehat{\mathcal{R}}) \text{ accepts the estimate } \widehat{R}\right) &\leq \sum_{\mathbf{j}\in[(d\log n)^*]^{d-1}} \mathbb{P}(X_{\mathbf{j}} \neq 0) \\ &\leq \frac{(d\log n + 1)^{d-1}}{20 \cdot 4^d \cdot (d\log n)^{d-1}} \\ &\leq \frac{1}{20 \cdot 2^d}.\end{aligned}$$

◀

▶ **Lemma 4.2.** *If* $\widehat{\mathcal{R}} \leq \frac{m_o(\mathcal{H}_o)}{4d\log n}$, $\mathbb{P}$(Verify-Estimate $(\mathcal{H}_o, \widehat{\mathcal{R}})$ *accepts the estimate* $\widehat{R}$) $\geq \frac{1}{2^d}$.

Proof. We will be done by showing the following. Verify-Estimate accepts with probability at least $1/5$ when the loop variables $j_1, \ldots, j_{d-1}$ respectively attain values $q_1, \ldots, q_{d-1}$ such that

$$|U_1(q_1)| > \frac{m_o(\mathcal{H}_o)}{2^{q_1+1}(d\log n + 1)}$$

and

$$|U_i((Q_i, A_i), q_i)| > \frac{2^{q_{i-1}}}{2^{q_i+1}(d\log n + 1)}$$

for all $i \in [d-1] \setminus \{1\}$. The existence of such j_is is evident from Claim 3.2. Let $\mathbf{q} = (q_1, \ldots, q_{d-1})$. Recall that $B_{i,\mathbf{q}} \subseteq U_i$ is the sample obtained when the loop variables $j_1, \ldots, j_{d-1}$ attain values $q_1, \ldots, q_{d-1}$, respectively. Let $\mathcal{E}_i, i \in [d-1]$, be the events defined as follows.

[6] Note that $\mathbf{j}$ is a vector but $X_{\mathbf{j}}$ is a scalar.

- $\mathcal{E}_1 \ : \ U_1(q_1) \cap B_{1,\mathbf{q}} \neq \emptyset$.
- $\mathcal{E}_i \ : \ U_j((Q_{j-1}, A_{j-1}), q_j) \cap B_{j,\mathbf{q}} \neq \emptyset$, where $2 \leq i \leq d-1$.

As noted earlier, Claim 3.2 says that for each $i \in [d-1]$, there exists a bucket in U_i having a *large* number of vertices contributing approximately the same number of hyperedges. The above events correspond to the nonempty intersection of vertices in heavy buckets corresponding to U_i and the sampled vertices $B_{i,\mathbf{j}}$, where $i \in [d-1]$. Observe that

$$\begin{aligned}
\mathbb{P}(\overline{\mathcal{E}_1}) &\leq \left(1 - \frac{2^{q_1}}{\widehat{\mathcal{R}}}\right)^{|U_1(q_1)|} \\
&\leq \exp\left(-\frac{2^{q_1}}{\widehat{\mathcal{R}}}\,|U_1(q_1)|\right) \\
&\leq \exp\left(-\frac{2^{q_1}}{\widehat{\mathcal{R}}} \cdot \frac{m_o(\mathcal{H}_o)}{2^{q_1+1}(d\log n + 1)}\right) \\
&\leq \exp(-1).
\end{aligned}$$

The last inequality uses the fact that $\widehat{\mathcal{R}} \leq \frac{m_o(\mathcal{H}_o)}{4d\log n}$, from the condition of the lemma. Assume that $\mathcal{E}_1$ occurs and $a_1 \in U_1(q_1) \cap B_{1,\mathbf{q}}$. We will bound the probability that $U_2(Q_1, A_1), q_2) \cap B_{2,\mathbf{q}} = \emptyset$, that is $\overline{\mathcal{E}_2}$. Note that, by Claim 3.2 (ii),

$$|U_2(Q_1, A_1), q_2)| \geq \frac{2^{q_1}}{2^{q_2+1}(d\log n + 1)}.$$

So,

$$\mathbb{P}\left(\overline{\mathcal{E}_2} \mid \mathcal{E}_1\right) \leq \left(1 - \frac{2^{q_2}}{2^{q_1}}\log n\right)^{|U_2(Q_1,A_1),q_2)|} \leq \exp(-1)$$

Assume that $\mathcal{E}_1, \ldots, \mathcal{E}_{i-1}$ hold, where $3 \leq i \in [d-1]$. Let $a_1 \in U_1(q_1)$ and $a_{i-1} \in A_{i-1}((Q_{i-2}, U_{i-2}), q_{i-1})$. We will bound the probability that $U_i((Q_{i-1}, A_{i-1}), q_i) \cap B_{i,\mathbf{q}} = \emptyset$, that is $\overline{\mathcal{E}_i}$. Note that

$$|U_i((Q_{i-1}, A_{i-1}), q_i)| \geq \frac{2^{q_{i-1}}}{2^{q_i+1}(d\log n + 1)}.$$

So, for $3 \leq i \in [d-1]$,

$$\mathbb{P}\left(\overline{\mathcal{E}_i} \mid \mathcal{E}_1, \ldots, \mathcal{E}_{i-1}\right) \leq \left(1 - \frac{2^{q_i}}{2^{q_{i-1}}}\log n\right)^{|U_i(Q_{i-1},A_{i-1}),q_i)|} \leq \exp(-1)$$

Assume that $\mathcal{E}_1, \ldots, \mathcal{E}_{d-1}$ hold. Let $a_1 \in U_1(q_1)$ and $a_{i-1} \in A_{i-1}((Q_{i-2}, A_{i-2}), q_{i-1})$ for all $i \in [d] \setminus \{1\}$. Let $S \subseteq U_d$ be the set of d-th vertex of the ordered hyperedges in $\mathcal{F}_o(\mathcal{H}_o)$ having u_j as the j-th vertex for all $j \in [d-1]$. Note that $|S| \geq 2^{q_{d-1}}$. Let $\mathcal{E}_d$ be the event that represents the fact $S \cap B_{d,\mathbf{q}} \neq \emptyset$. So,

$$\mathbb{P}(\overline{\mathcal{E}_d} \mid \mathcal{E}_1, \ldots, \mathcal{E}_{d-1}) \leq \left(1 - \frac{1}{2^{q_{d-1}}}\right)^{q_{d-1}} \leq \exp(-1)$$

Observe that Verify-Estimate accepts if $m(B_{1,\mathbf{q}}, \ldots, B_{d,\mathbf{q}}) \neq 0$. Also,

$$m_o(B_{1,\mathbf{q}}, \ldots, B_{d,\mathbf{q}}) \neq 0 \text{ if } \bigcap_{i=1}^{d} \mathcal{E}_i \text{ occurs.}$$

Hence,

$$\begin{aligned}\mathbb{P}(\textsc{Verify-Estimate}\ (\mathcal{H}_o, \widehat{\mathcal{R}})\ \text{accepts}) &\geq \mathbb{P}\left(\bigcap_{i=1}^{d} \mathcal{E}_i\right)\\ &= \mathbb{P}(\mathcal{E}_1)\prod_{i=2}^{d}\mathbb{P}\left(\mathcal{E}_i \;\middle|\; \bigcap_{j=1}^{i-1}\mathcal{E}_j\right)\\ &> \left(1-\frac{1}{e}\right)^d\\ &> \frac{1}{2^d}.\end{aligned}$$

◀

Now, we will prove Lemma 3.1 that will be based on Algorithm 2.

Algorithm 2 Rough Estimation($\mathcal{H}_o(U, \mathcal{F}_o)$).

Input: CID$_2^o$ query access to a d-uniform hypergraph $\mathcal{H}_o(U, \mathcal{F}_o)$.
Output: An estimate $\widehat{m}_o$ for $m_o = m_o(\mathcal{H}_o)$.
for ($\widehat{\mathcal{R}} = n^d, n^d/2, \ldots, 1$) **do**
  Repeat Verify-Estimate $(\mathcal{H}_o, \widehat{\mathcal{R}})$ for $\Gamma = d \cdot 4^d \cdot 2000 \log n$ times. If more than $\frac{\Gamma}{10\cdot 2^d}$ Verify-Estimate accepts, then output $\widehat{m}_o = \frac{\widehat{\mathcal{R}}}{d^{d-2}\cdot 2^d \cdot (\log n)^{d-2}}$.
end

Proof of Lemma 3.1. Note that an execution of Rough Estimation for a particular $\widehat{\mathcal{R}}$ repeats Verify-Estimate for $\Gamma = d \cdot 4^d \cdot 2000 \log n$ times and gives output $\widehat{\mathcal{R}}$ if more than $\frac{\Gamma}{10\cdot 2^d}$ Verify-Estimate accepts. For a particular $\widehat{\mathcal{R}}$, let X_i be the indicator random variable such that $X_i = 1$ if and only if the i-th execution of Verify-Estimate accepts. Also take $X = \sum_{i=1}^{\Gamma} X_i$. Rough Estimation gives output $\widehat{\mathcal{R}}$ if $X > \frac{\Gamma}{10\cdot 2^d}$.

Consider the execution of Rough Estimation for a particular $\widehat{\mathcal{R}}$. If $\widehat{\mathcal{R}} \geq 20d^{2d-3}4^d \cdot m_o(\mathcal{H}_o)\cdot \log^{2d-3} n$, then we first show that Rough Estimation does not accept with high probability. Recall Lemma 4.1. If $\widehat{\mathcal{R}} \geq 20d^{2d-3}4^d \cdot m_o(\mathcal{H}_o)\log^{2d-3} n$, $\mathbb{P}(X_i = 1) \leq \frac{1}{20\cdot 2^d}$ and hence $\mathbb{E}[X] \leq \frac{\Gamma}{20\cdot 2^d}$. By using Chernoff-Hoeffding's inequality (See Lemma A.2 (i) in Section A),

$$\mathbb{P}\left(X > \frac{\Gamma}{10\cdot 2^d}\right) = \mathbb{P}\left(X > \frac{\Gamma}{20\cdot 2^d} + \frac{\Gamma}{20\cdot 2^d}\right) \leq \frac{1}{n^{10d}}$$

Using the union bound for all $\widehat{\mathcal{R}}$, the probability that Rough Estimation outputs some $\widehat{m}_o = \frac{\widehat{\mathcal{R}}}{d^{d-2}\cdot 2^d}$ such that $\widehat{\mathcal{R}} \geq 20d^{2d-3}4^d \cdot m_o(\mathcal{H}_o)\log^{2d-3} n$, is at most $\frac{d\log n}{n^{10}}$. Now consider the instance when the for loop in the algorithm Rough Estimation executes for a $\widehat{\mathcal{R}}$ such that $\widehat{\mathcal{R}} \leq \frac{m_o(\mathcal{H}_o)}{4d\log n}$. In this situation, $\mathbb{P}(X_i = 1) \geq \frac{1}{2^d}$. So, $\mathbb{E}[X] \geq \frac{\Gamma}{2^d}$. By using Chernoff-Hoeffding's inequality (See Lemma A.2 (ii) in Section A),

$$\mathbb{P}\left(X \leq \frac{\Gamma}{10\cdot 2^d}\right) \leq \mathbb{P}\left(X < \frac{\Gamma}{2^d} - \frac{4}{5}\cdot\frac{\Gamma}{2^d}\right) \leq \frac{1}{n^{100d}}$$

By using the union bound for all $\widehat{\mathcal{R}}$, the probability that Rough Estimation outputs some $\widehat{m}_o = \frac{\widehat{\mathcal{R}}}{d^{d-2}\cdot 2^d}$ such that $\widehat{\mathcal{R}} \leq \frac{m_o(\mathcal{H}_o)}{4d\log n}$, is at most $\frac{d\log n}{n^{100d}}$. Observe that, the probability that Rough Estimation outputs some $\widehat{m}_o = \frac{\widehat{\mathcal{R}}}{d^{d-2}\cdot 2^d}$ such that $\widehat{\mathcal{R}} \geq 20d^{2d-3}4^d m_o(\mathcal{H}_o)\log^{2d-3} n$ or $\widehat{\mathcal{R}} \leq \frac{m_o(\mathcal{H}_o)}{4d\log n}$, is at most

$$\frac{d\log n}{n^{10d}} + \frac{d\log n}{n^{100d}} \leq \frac{1}{n^{8d}}.$$

Putting everything together, Rough Estimation gives some $\widehat{m}_o = \frac{\widehat{\mathcal{R}}}{d^{d-2}\cdot 2^d\cdot(\log n)^{d-2}}$ as the output with probability at least $1 - \frac{1}{n^{8d}}$ satisfying

$$\frac{m_o(\mathcal{H}_o)}{8d^{d-1}2^d\log^{d-1} n} \leq \widehat{m}_o \leq 20d^{d-1}2^d \cdot m_o(\mathcal{H}_o)\log^{d-1} n$$

From the pseudocode of Verify-Estimate (Algorithm 1), we call for CID_2 queries only at line number 12. In the worst case, Verify-Estimate executes line number 12 for each $\mathbf{j} \in [(d\log n)^*]$. That is, the query complexity of Verify-Estimate is $\mathcal{O}(\log^{d-1} n)$. From the description of Rough Estimation, Rough Estimation calls Verify-Estimate $\mathcal{O}_d(\log n)$ times for each choice of $\widehat{R}$. Hence, Rough Estimation makes $\mathcal{O}_d(\log^{d+1} n)$ CID_2^o queries. ◀

5 Proof of Theorem 1.4

Before getting into the reasons *why Theorem 1.4 follows from Theorem 1.3*, let us first review the algorithms for d-Hyperedge-Estimation and d-Hyperedge-Sample by Dell et al. [9].

Overview of Dell et al. [9]

Dell et al.'s algorithm for d-Hyperedge-Sample make repeated calls to d-Hyperedge-Estimation. Their algorithm for d-Hyperedge-Estimation calls mainly three subroutines over $\mathcal{O}_d(\log n)$ iterations: Coarse, Halving, and Trim. Halving and Trim calls Coarse repeatedly. So, Coarse is the main building block for their algorithms for d-Hyperedge-Estimation and d-Hyperedge-Sample.

Coarse algorithm

It estimates the number of hyperedges in the hypergraph up to polylog factors by using polylog queries. The result is formally stated as follows, see [9, Sec. 4].

▶ **Lemma 5.1** (Coarse Algorithm by Dell et al. [9]). *There exists an algorithm* Coarse*, that has* CID *query access to a hypergraph* $\mathcal{H}(U,\mathcal{F})$*, makes* $\mathcal{O}_d\left(\log^{2d+3} n\right)$ CID *queries, and finds* $\widehat{m}$ *satisfying*

$$\Omega_d\left(\frac{1}{\log^d n}\right) \leq \frac{\widehat{m}}{m} \leq \mathcal{O}_d\left(\log^d n\right)$$

with probability at least $1 - 1/n^{\Omega(d)}$.

▶ Remark. The objective of Coarse algorithm by Dell et al. is essentially same as that our Rough Estimation algorithm. Both of them can estimate the number of hyperedges in any induced subhypergrah. However, note that Rough Estimation (as stated in Theorem 1.3) has better approximation guarantee and better query complexity than that of Coarse algorithm of Dell et al. (as stated in Lemma 5.1).

The framework of Dell et al. implies that the query complexity of d-Hyperedge-Estimation and d-Hyperedge-Sample can be expressed by the approximation guarantee and the query complexity of the Coarse algorithm. This is formally stated as follows:

▶ **Lemma 5.2** (d-HYPEREDGE-ESTIMATION and d-HYPEREDGE-SAMPLE in terms of quality of COARSE algorithm [9]). *Let there exists an algorithm* COARSE*, that has* CID *query access to a hypergraph $\mathcal{H}(U,\mathcal{F})$, makes q* CID *queries, and finds $\widehat{m}$ satisfying $\frac{1}{b} \leq \frac{\widehat{m}}{m} \leq b$ with probability at least $1-1/n^{\Omega(d)}$. Then*

(i) *d-*HYPEREDGE-ESTIMATION *can be solved by using*

$$\mathcal{O}_d\left(\log^2 n\left(\log nb + \frac{b^2\log^2 n}{\varepsilon^2}\right)q\right)$$

CID *queries.*

(ii) *d-*HYPEREDGE-SAMPLE *can be solved by using*

$$\mathcal{O}_d\left(\log^6 n\left(\log nb + \frac{b^2\log^2 n}{\varepsilon^2}\right)q\right)$$

CID *queries.*

Why Theorem 1.4 follows from Theorem 1.3?

Observe that we get Proposition 1.2 (the result of Dell et al.) from Lemma 5.1 by substituting $b = \mathcal{O}_d\left(\log^d n\right)$ and $q = \mathcal{O}_d\left(\log^{2d+3} n\right)$ in Lemma 5.2. In Theorem 1.4 we improve on the Proposition 1.2 by using our main result (Theorem 1.3), and substituting $b = \mathcal{O}_d\left(\log^{d-1} n\right)$ and $q = \mathcal{O}_d\left(\log^{d+2} n\right)$ in Lemma 5.2.

The main reason we get an improved query complexity for hyperedge estimation in Theorem 1.4 as compared to Dell et al. (Proposition 5.2) is our ROUGH ESTIMATION algorithm is an improvement over the COARSE algorithm of Dell et al. [9] in terms of approximation guarantee as well as query complexity.

How our ROUGH ESTIMATION improves over COARSE of Dell et al. [9]?

At a very high level, the frameworks of our ROUGH ESTIMATION algorithm and that of Dell et al.'s COARSE algorithm might look similar, but the main ideas involved are different. Our ROUGH ESTIMATION (as stated in Lemma 3.1) directly deals with the hypergraph (though the ordered one) and makes use of CID_2^o queries. Note that each CID_2^o query can be simulated by using $\mathcal{O}_d(\log n)$ CID queries. However, COARSE algorithm of Dell et al. considers $\mathcal{O}_d(\log n)$ independent random d-partite hypergraphs by partitioning the vertex set into d parts uniformly at random, works on the d-partite hypergraphs, and reports the median, of the $\mathcal{O}_d(\log n)$ outputs corresponding to random d-partite subhypergrahs, as the final output. So, there is $\mathcal{O}_d(\log n)$ blowup in both our ROUGH ESTIMATION algorithm and Dell et al.'s COARSE algorithm, though the reasons behind the blowups are different.

Our ROUGH ESTIMATION calls repeatedly ($\mathcal{O}_d(\log n)$ times) VERIFY ESTIMATE for each guess, where the total number of guesses is $\mathcal{O}_d(\log n)$. In the COARSE algorithm, Dell et al. uses repeated calls $\left(\mathcal{O}_d\left(\log^{d+1} n\right)\right)$ times to an analogous routine of our VERIFY ESTIMATE, which they name VERIFY GUESS, $\mathcal{O}_d(\log n)$ times. Their VERIFY GUESS has the following criteria for any guess M:

- If $M \geq \frac{d^d \log^{2d} n}{2^{3d-1}} m$, VERIFY GUESS accepts M with probability at most p;
- If $M \leq m$, VERIFY GUESS accepts M with probability at least $2p$;
- It makes $\mathcal{O}_d\left(\log^d n\right)$ CID queries.

Recall that the number of CID_2 queries made by each call to Verify Estimate is $\mathcal{O}_d(\log^{d-1} n)$, that is, $\mathcal{O}_d\left(\log^d n\right)$ CID queries. So, in terms of the number of CID queries, both our Rough Estimation and Coarse of Dell et al. have the same complexity.

The probability p in Verify Guess of Dell et al. [9] satisfies $p \approx_d \frac{1}{\log^d n}$, where $\approx_d$ is used suppress the terms involving d. So, for each guess M , their Coarse algorithm has to call $\mathcal{O}_d\left(\frac{1}{p}\log n\right) = \mathcal{O}_d\left(\log^{d+1} n\right)$ times to distinguish decide whether it is the case $M \leq m$ or $M \geq \frac{d^d \log^{2d} n}{2^{3d-1}} m$, with a probability at least $1 - 1/n^{\Omega(d)}$. So, the total number of queries made by the Coarse algorithm of Dell et al. [9] is

$$\mathcal{O}_d(\log n) \cdot \mathcal{O}_d(\log n) \cdot \mathcal{O}_d\left(\log^{d+1} n\right) \cdot \mathcal{O}_d\left(\log^d n\right) = \mathcal{O}_d\left(\log^{2d+3} n\right).$$

The first $\mathcal{O}_d(\log n)$ term is due to the blow up incurred to convert original hypergraph to d-partite hypergraph, the second $\mathcal{O}_d(\log n)$ term is due to the number of guesses for m, the third $\mathcal{O}_d\left(\log^{d+1} n\right)$ term is the number of times Coarse calls Verify Guess, and the last term $\mathcal{O}_d\left(\log^d n\right)$ is the number of CID queries made by each call to Verify Guess.

As it can be observed from Lemmas 4.1 and 4.2, p in our case (Verify Estimate) is $\Omega_d(1)$. So, it is enough for Rough Estimation to call Verify Estimate only $\mathcal{O}_d(\log n)$ times. Therefore, the number of CID queries made by our Rough Estimation is

$$\mathcal{O}_d(\log n) \cdot \mathcal{O}_d(\log n) \cdot \mathcal{O}_d(\log^{d-1} n) \cdot \mathcal{O}_d(\log n) = \mathcal{O}_d(\log^{d+2} n).$$

In the above expression, the first $\mathcal{O}_d(\log n)$ term is due to the number of guesses for m, the second $\mathcal{O}_d(\log n)$ term is the number of times Rough Estimation calls Verify Estimate, the third $\mathcal{O}\left(\log^{d-1} n\right)$ term is the number of CID_2 queries made by each call to Verify Estimate, and the last $\mathcal{O}_d(\log n)$ term is the number of CID queries needed to simulate a CID_2 query with probability at least $1 - 1/n^{\Omega(d)}$.

We do the improvement in approximation guarantee as well as query complexity in Rough Estimation algorithm (as stated in Theorem 1.3), as compared to Coarse algorithm of Dell et al. [9] (as stated in Lemma 5.1), by a careful analysis of the intersection pattern of the hypergraphs and setting the sampling probability parameters in Verify Estimate (Algorithm 1) algorithm in a nontrivial way, which is evident from the description of Algorithm 1 and its analysis.

References

1 Paul Beame, Sariel Har-Peled, Sivaramakrishnan Natarajan Ramamoorthy, Cyrus Rashtchian, and Makrand Sinha. Edge Estimation with Independent Set Oracles. In *Proceedings of the 9th Innovations in Theoretical Computer Science Conference, ITCS*, volume 94, pages 38:1–38:21, 2018.

2 Paul Beame, Sariel Har-Peled, Sivaramakrishnan Natarajan Ramamoorthy, Cyrus Rashtchian, and Makrand Sinha. Edge Estimation with Independent Set Oracles. *ACM Trans. Algorithms*, 16(4):52:1–52:27, 2020.

3 Anup Bhattacharya, Arijit Bishnu, Arijit Ghosh, and Gopinath Mishra. Triangle Estimation Using Tripartite Independent Set Queries. In *Proceedings of the 30th International Symposium on Algorithms and Computation, ISAAC*, volume 149, pages 19:1–19:17, 2019.

4 Anup Bhattacharya, Arijit Bishnu, Arijit Ghosh, and Gopinath Mishra. On Triangle Estimation Using Tripartite Independent Set Queries. *Theory Comput. Syst.*, 65(8):1165–1192, 2021.

5 Anup Bhattacharya, Arijit Bishnu, Arijit Ghosh, and Gopinath Mishra. Faster Algorithms for Estimating and Sampling using Colorful Decision Oracle, 2022. `arXiv:2201.04975`.

6 Arijit Bishnu, Arijit Ghosh, Sudeshna Kolay, Gopinath Mishra, and Saket Saurabh. Parameterized Query Complexity of Hitting Set Using Stability of Sunflowers. In *Proceedings of the 29th International Symposium on Algorithms and Computation, ISAAC*, volume 123, pages 25:1–25:12, 2018.

7 Holger Dell and John Lapinskas. Fine-Grained Reductions from Approximate Counting to Decision. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC*, pages 281–288, 2018.

8 Holger Dell and John Lapinskas. Fine-Grained Reductions from Approximate Counting to Decision. *ACM Trans. Comput. Theory*, 13(2):8:1–8:24, 2021.

9 Holger Dell, John Lapinskas, and Kitty Meeks. Approximately counting and sampling small witnesses using a colourful decision oracle. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 2201–2211, 2020.

10 Devdatt P. Dubhashi and Alessandro Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009.

11 Talya Eden, Amit Levi, Dana Ron, and C. Seshadhri. Approximately counting triangles in sublinear time. *SIAM J. Comput.*, 46(5):1603–1646, 2017. doi:10.1137/15M1054389.

12 Talya Eden, Dana Ron, and C. Seshadhri. On Approximating the Number of k-Cliques in Sublinear Time. *SIAM J. Comput.*, 49(4):747–771, 2020.

13 Uriel Feige. On Sums of Independent Random Variables with Unbounded Variance and Estimating the Average Degree in a Graph. *SIAM J. Comput.*, 35(4):964–984, 2006.

14 Oded Goldreich and Dana Ron. Approximating Average Parameters of Graphs. *Random Struct. Algorithms*, 32(4):473–493, 2008.

A Some probability results

▶ **Lemma A.1** (Chernoff-Hoeffding bound [10]). *Let $X_1, \ldots, X_n$ be independent random variables such that $X_i \in [0,1]$. For $X = \sum_{i=1}^{n} X_i$ and $\mu = \mathbb{E}[X]$, the followings hold for any $0 \leq \delta \leq 1$.*

$$\mathbb{P}(|X - \mu| \geq \delta\mu) \leq 2\exp\left(-\mu\delta^2/3\right)$$

▶ **Lemma A.2** (Chernoff-Hoeffding bound [10]). *Let $X_1, \ldots, X_n$ be independent random variables such that $X_i \in [0,1]$. For $X = \sum_{i=1}^{n} X_i$ and $\mu_l \leq \mathbb{E}[X] \leq \mu_h$, the followings hold for any $\delta > 0$.*

(i) $\mathbb{P}\left(X > \mu_h + \delta\right) \leq \exp\left(-2\delta^2/n\right)$.

(ii) $\mathbb{P}\left(X < \mu_l - \delta\right) \leq \exp\left(-2\delta^2/n\right)$.

B Oracle definitions

▶ **Definition B.1** (Independent set query (IS) [1]). *Given a subset A of the vertex set V of a graph $G(V,E)$,* IS *query answers whether A is an independent set.*

▶ **Definition B.2** (Bipartite independent set oracle (BIS) [1]). *Given two disjoint subsets A, B of the vertex set V of a graph $G(V,E)$,* BIS *query reports whether there exists an edge having endpoints in both A and B.*

▶ **Definition B.3** (Tripartite independent set oracle (TIS) [3]). *Given three disjoint subsets A, B, C of the vertex set V of a graph $G(V,E)$, the* TIS *oracle reports whether there exists a triangle having endpoints in A, B and C.*

▶ **Definition B.4** (Generalized d-partite independent set oracle (CID) [6]). *Given d pairwise disjoint subsets of vertices $A_1, \ldots, A_d \subseteq U(\mathcal{H})$ of a hypergraph $\mathcal{H}$ as input,* CID *query answers whether $m(A_1, \ldots, A_d) \neq 0$, where $m(A_1, \ldots, A_d)$ denotes the number of hyperedges in $\mathcal{H}$ having exactly one vertex in each A_i, $\forall i \in \{1, 2, \ldots, d\}$.*

▶ **Definition B.5** (CID_1 oracle). *Given s pairwise disjoint subsets of vertices $A_1, \ldots, A_s \subseteq U(\mathcal{H})$ of a hypergraph $\mathcal{H}$ and $a_1, \ldots, a_s \in [d]$ such that $\sum_{i=1}^{s} a_i = d$,* CID_1 *query on input $A_1^{[a_1]}, A_2^{[a_2]}, \cdots, A_s^{[a_s]}$ answers whether $m(A_1^{[a_1]}, \ldots, A_s^{[a_s]}) \neq 0$.*

▶ **Definition B.6** (CID_2 oracle). *Given any d subsets of vertices $A_1, \ldots, A_d \subseteq U(\mathcal{H})$ of a hypergraph $\mathcal{H}$,* CID_2 *query on input $A_1, \ldots, A_d$ answers whether $m(A_1, \ldots, A_d) \neq 0$.*

▶ **Definition B.7** (CID_2^o oracle). *Given any d subsets of vertices $A_1, \ldots, A_d \subseteq U(\mathcal{H}_o)$ of an ordered hypergraph $\mathcal{H}_o$,* CID_2^o *query on input $A_1, \ldots, A_d$ answers* Yes *if and only if $m_o(A_1, \ldots, A_d) \neq 0$.*