Asymptotically Optimal Locally Private Heavy Hitters via Parameterized Sketches

Hao Wu
School of Computing & Information Systems
The University of Melbourne

Anthony Wirth
School of Computing & Information Systems
The University of Melbourne

Abstract

We study the frequency estimation problem under the local differential privacy model. Frequency estimation is a fundamental computational question, and differential privacy has become the de-facto standard, with the local version (LDP) affording even greater protection. On large input domains, sketching methods and hierarchical search methods are commonly and successfully, in practice, applied for reducing the size of the domain, and for identifying frequent elements. It is therefore of interest whether the current theoretical analysis of such algorithms is tight, or whether we can obtain algorithms in a similar vein that achieve optimal error guarantee.

We introduce two algorithms for LDP frequency estimation. One solves the fundamental frequency oracle problem; the other solves the well-known heavy hitters identification problem. As a function of failure probability, β, the former achieves optimal worst-case estimation error for every β; the latter is optimal when β is at least inverse polynomial in n, the number of users. In each algorithm, server running time and memory usage are $\tilde{O}(n)$ and $\tilde{O}(\sqrt{n})$, respectively, while user running time and memory usage are both $\tilde{O}(1)$. Our frequency-oracle algorithm achieves lower estimation error than Bassily et al. (NeurIPS 2017). On the other hand, our heavy hitters identification method improves the worst-case error of **TreeHist** (ibid) by a factor of $\Omega(\sqrt{\log n})$; it avoids invoking error-correcting codes, known to be theoretically powerful, but yet to be implemented efficiently.

1 Introduction

Frequency estimation is a fundamental computation task, widely applied in data mining and machine learning, e.g., learning users' preferences (Erlingsson et al., 2014), uncovering commonly used phrases (Apple, 2017), and finding popular URLs (Fanti et al., 2016). We expect entities that collect such data to respect their users' privacy, and there are increasing stringent regulations (Voigt and Von dem Bussche, 2017). How can we infer and estimate frequency, and thus improve users' experience, without sacrificing personal privacy?

In answering such questions, local differential privacy (**LDP**) becomes a popular data collection model for providing user level privacy protection (Erlingsson et al., 2014; Fanti et al., 2016; Apple, 2017; Tang et al., 2017; Ding et al., 2017). In this model, there is a server and a set $\mathcal{U}$ of n users, each holding an element from some domain $\mathcal{D}$ of size d. No user $u \in \mathcal{U}$ wants to share their data $v^{(u)} \in \mathcal{D}$ directly with the server. To protect sensitive personal information, they run a local randomizer $\mathcal{A}^{(u)}$ to perturb their data. The server collects only the perturbed data. Formally, the algorithm $A^{(u)}$ is called *ε-local differentially private* (ε-**LDP**) if its output distribution varies little with the input, as defined thus.

Definition 1.1 (ε-**LDP** (Dwork and Roth, 2014))**.** Let $\mathcal{A} : \mathcal{D} \to \mathcal{Y}$ be a randomized algorithm mapping an element in $\mathcal{D}$ to $\mathcal{Y}$. We say $\mathcal{A}$ is ε-local differentially private if for all $v, v' \in \mathcal{D}$ and all (measurable) $Y \subseteq \mathcal{Y}$,

$$\Pr[\mathcal{A}(v) \in Y] \le e^{\varepsilon} \cdot \Pr[\mathcal{A}(v') \in Y] .$$

Aligned with prior art (Bassily and Smith, 2015; Bassily et al., 2017; Bun et al., 2019; Cormode et al., 2021), in this work, we study two closely related, but distinct, functionalities of frequency estimation under the **LDP** model: the frequency oracle and the succinct histogram. The relevant parameters are the error threshold, λ, and the failure probability, β:

Definition 1.2 (Frequency Oracle)**.** A frequency oracle, denoted as **FO**, is an algorithm that provides for

Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151.

every $v \in \mathcal{D}$, an estimated $\hat{f}_{\mathcal{U}}[v]$ of the frequency of v, denoted as $f_{\mathcal{U}}[v] \doteq |\{u \in \mathcal{U} : v^{(u)} = v\}|$, such that $\Pr[|\hat{f}_{\mathcal{U}}[v] - f_{\mathcal{U}}[v]| \geq \lambda] \leq \beta$.

Definition 1.3 (Succinct Histogram)**.** A succinct histogram, denoted as **S-Hist**, is a set of (element, estimate) pairs $\subseteq \mathcal{D} \times \mathbb{R}$, of size $O(n/\lambda)$, such that with probability at least $1 - \beta$: i) $\forall v \in \mathcal{D}$, if $f_{\mathcal{U}}[v] \geq \lambda$, $(v, \hat{f}_{\mathcal{U}}[v])$ belongs to the set; ii) and if $(v, \hat{f}_{\mathcal{U}}[v])$ is in the set, then $|f_{\mathcal{U}}[v] - \hat{f}_{\mathcal{U}}[v]| \leq \lambda$.

Each element in the Succinct Histogram set is called a *heavy hitter*. For a fixed $\varepsilon \leq 1$, the goal of algorithm design for both problems in **LDP** model is to minimize the error threshold λ, while also limiting server/user running time, memory usage, and communication cost.

A number of frequency oracle algorithms (Warner, 1965; Erlingsson et al., 2014; Bassily and Smith, 2015; Bassily et al., 2017; Wang et al., 2017) have been proposed (see Cormode et al. (2021) for a recent survey). These algorithms achieve error $O((1/\varepsilon) \cdot \sqrt{n \ln(1/\beta)})$. They have running time, or memory usage that scale linearly with d, the size of the data domain, and work well when it is small. The heavy hitters can be discovered by querying the frequencies of all elements in the domain $\mathcal{D}$. Via union bound, this achieves error $O((1/\varepsilon) \cdot \sqrt{n \ln(d/\beta)})$. It can be shown that these error guarantees are optimal (Bun et al., 2019). However, consider the scale of modern applications, e.g., finding popular URLs with length up to 20 characters[1] (Fanti et al., 2016), which results in a domain of size larger than 10^{36}.

Sketching methods for reducing the size of the data domain, and *hierarchical searching methods* for avoiding inspecting the frequency of each element, are well known. The former applies hash functions to map elements from the original domain to a smaller one; the latter views elements as strings defined over some alphabet, and identifies the heavy hitters by one or a few characters each time. Due to their simplicity, they are widely applied in designing frequency estimation algorithms, and heuristics are proposed to improve their performance (Bassily et al., 2017; Apple, 2017; Fanti et al., 2016; Bassily et al., 2020; Wang et al., 2018, 2017; Cormode et al., 2021). These implementations perform well in practice.

The best known error guarantees of frequency estimation algorithms based on the *sketching* and *hierarchical searching methods* are provided by the seminal work (Bassily et al., 2020). The frequency oracles, **FreqOracle** and **Hashtogram** (Bassily et al., 2020), guarantee only an error of $O((1/\varepsilon) \cdot \sqrt{n \ln(n/\beta)})$. The succinct histogram algorithm, **TreeHist** (Bassily et al., 2020), guarantees an error of $O((1/\varepsilon) \cdot \sqrt{n \cdot (\ln d) \cdot \ln(n/\beta)})$. These algorithms exhibit low time complexity and memory usage: with server running time $\tilde{O}(n)$ and memory usage $\tilde{O}(\sqrt{n})$. But the error guarantees are sub-optimal.

> **Research Question:** Are the theoretical error guarantees of the algorithms based on sketching and hierarchy methods best possible, or can we obtain algorithms of this type that achieve optimal error guarantee?

There is another line of research for succinct histogram that relies on error-correcting codes. Bassily and Smith (Bassily and Smith, 2015) were the pioneers, with **PROT-S-Hist**, which **Bitstogram** (Bassily et al., 2020) subsequently improved upon. This culminates in the work of **PrivateExpanderSketch** (Bun et al., 2019) that achieves the optimal error guarantee $O((1/\varepsilon) \cdot \sqrt{n \ln(d/\beta)})$. But due to the sophistication of error-correcting codes, none of these methods has been implemented or significantly deployed. Indeed, the original paper (Bassily et al., 2020) that proposed both **TreeHist** and **Bitstogram** only implemented **TreeHist**. Therefore, there is a gap between the error guarantees of the theoretically best algorithm, and the ones deployed in practice. Determining whether we can bridge the gap answers our Research Question.

1.1 Our Contributions

Our work provides positive answers to the questions. In particular, we: (1) design a frequency oracle, **HadaOracle**, based on *sketching method* with optimal error guarantee; (2) design a succinct histogram algorithm, **HadaHeavy**, based on *hierarchical search* that achieves optimal error guarantee under mild assumption of the failure probability.

We introduce the martingale method into the analysis of the *sketching method*. We prove that, when the proper sketch is chosen, it can be incorporated into a family of frequency oracles to reduce running time and memory, while maintaining the frequency oracles' error guarantee. This leads to **HadaOracle** with optimal error $O((1/\varepsilon) \cdot \sqrt{n \ln(1/\beta)})$. Based on the **HadaOracle**, we develop a *hierarchical search* algorithm, **HadaHeavy**, that explores a large number of elements at each search step, and achieves error $O((1/\varepsilon) \cdot \sqrt{n \cdot (\ln d) \cdot (1 + (\ln(1/\beta) \ / \ln n))})$. Consistent with the theory community's view of an algorithm that succeeds with high probability, if the failure probability, β, is inverse polynomial, i.e., $\beta = 1/n^{O(1)}$, the error matches the lower bound (Bun et al., 2019). All these algorithms have running time $\tilde{O}(n)$ and memory usage $\tilde{O}(\sqrt{n})$. Table 1 summarizes the comparisons.

[1] Valid URL characters include digits (0-9), letters(A-Z, a-z), and a few special characters (”-” , ”.” , ”_” , ”~”).

	Performance Metric	Server Time	Server Mem	Worst-Case Error	Lower Bound
FO	**HadaOracle**	$\tilde{O}(n)$	$\tilde{O}(\sqrt{n})$	$O\left(\frac{1}{\varepsilon}\sqrt{n \cdot \ln\frac{1}{\beta}}\right)$	$O\left(\frac{1}{\varepsilon}\sqrt{n \cdot \ln\frac{1}{\beta}}\right)$
	HRR (Nguyên et al., 2016; Cormode et al., 2019)	$\tilde{O}(d)$	$\tilde{O}(d)$	$O\left(\frac{1}{\varepsilon}\sqrt{n \cdot \ln\frac{1}{\beta}}\right)$	
	FreqOracle (Bassily et al., 2017)	$\tilde{O}(n)$	$\tilde{O}(\sqrt{n})$	$O\left(\frac{1}{\varepsilon}\sqrt{n \cdot \ln\frac{n}{\beta}}\right)$	
	Hashtogram (Bassily et al., 2017)	$\tilde{O}(n)$	$\tilde{O}(\sqrt{n})$	$O\left(\frac{1}{\varepsilon}\sqrt{n \cdot \ln\frac{n}{\beta}}\right)$	
S-Hist	**HadaHeavy**	$\tilde{O}(n)$	$\tilde{O}(\sqrt{n})$	$O\left(\frac{1}{\varepsilon}\sqrt{n \cdot (\ln d) \cdot \left(1 + \frac{\ln(1/\beta)}{\ln n}\right)}\right)$	$O\left(\frac{1}{\varepsilon}\sqrt{n \cdot \ln\frac{d}{\beta}}\right)$
	TreeHist (Bassily et al., 2017)	$\tilde{O}(n)$	$\tilde{O}(\sqrt{n})$	$O\left(\frac{1}{\varepsilon}\sqrt{n \cdot (\ln d) \cdot \ln\frac{n}{\beta}}\right)$	
	PrivateExpanderSketch (Bun et al., 2019)	$\tilde{O}(n)$	$\tilde{O}(\sqrt{n})$	$O\left(\frac{1}{\varepsilon}\sqrt{n \cdot \ln\frac{d}{\beta}}\right)$	
	Bitstogram (Bassily et al., 2017)	$\tilde{O}(n)$	$\tilde{O}(\sqrt{n})$	$O\left(\frac{1}{\varepsilon}\sqrt{n \cdot (\ln\frac{d}{\beta}) \cdot \ln\frac{1}{\beta}}\right)$	

Table 1: Comparison of our frequency oracle (**HadaOracle**) and succinct histogram (**HadaHeavy**) algorithms with the state-of-the-art, where 'Mem' stands for 'Memory'. For all algorithms, each user has $\tilde{O}(1)$ memory, takes $\tilde{O}(1)$ running time, requires $\tilde{O}(1)$ public randomness, and reports $O(1)$ bits to the server.

2 Preliminaries

2.1 Hadamard Randomized Response

Our algorithms invoke the frequency oracle, named **HRR** (Nguyên et al., 2016; Cormode et al., 2019), as a subroutine.

Fact 2.1 (Algorithm **HRR** (Nguyên et al., 2016; Cormode et al., 2019))**.** *Let $\mathcal{U}$ be a set users each holding an element from some finite domain $\mathcal{D}$. There exists an ε-locally differentially private frequency oracle, **HRR**, such that the following holds. Fix some query element $v \in \mathcal{D}$ for **HRR**. With probability at least $1-\beta'$, **HRR** returns a frequency estimate $\hat{f}_{\mathcal{U}}[v]$ satisfying*

$$\left|\hat{f}_{\mathcal{U}}[v] - f_{\mathcal{U}}[v]\right| \in O\left((1/\varepsilon) \cdot \sqrt{|\mathcal{U}| \cdot \ln(1/\beta')}\right).$$

Each user in $\mathcal{U}$ requires $\tilde{O}(1)$ memory, takes $\tilde{O}(1)$ running time and reports only 1 bit to the server. The server processes the reports in $\tilde{O}(|\mathcal{U}| + |\mathcal{D}|)$ time and $O(|\mathcal{D}|)$ memory, and answers a query in $\tilde{O}(1)$ time. The $\tilde{O}$ notation hides logarithmic factors in $|\mathcal{U}|$, $|\mathcal{D}|$ and $1/\beta'$.

The Appendix includes a proof of this fact.

2.2 Lower Bounds

Bun et al. (2019) provide a lower bound for the succinct histogram problem.

Fact 2.2 (Bun et al. (2019))**.** *Let $\varepsilon \in O(1)$. Every ε-**LDP** algorithm for estimating the frequencies of all elements from $\mathcal{D}$, must have, with probability at least $1-\beta$,*

$$\max_{v \in \mathcal{D}} \left|\hat{f}_{\mathcal{U}}[v] - f_{\mathcal{U}}[v]\right| \in \Omega\left((1/\varepsilon) \cdot \sqrt{|\mathcal{U}| \cdot \ln(|\mathcal{D}|/\beta)}\right).$$

We can obtain a lower bound for the frequency oracles, via a union bound argument, with $\beta' = \beta/|\mathcal{D}|$.

Corollary 2.3. *Let $\varepsilon \in O(1)$. Every ε-**LDP** frequency oracle algorithm achieving estimation error λ with probability at least $1-\beta'$ must have*

$$\lambda \in \Omega\left((1/\varepsilon) \cdot \sqrt{|\mathcal{U}| \cdot \ln(1/\beta')}\right).$$

The Appendix includes a proof of this corollary.

3 Frequency Oracle

In this section, to reduce running time and memory usage, we show a framework for incorporating sketching methods into **LDP** frequency oracles that satisfy relevant conditions. When combined with **HRR**, this gives arise to a frequency oracle with a state-of-the-art theoretical guarantee.

Suppose $\mathcal{A}_{oracle}$ is an ε-**LDP** frequency oracle with:

- Server running time: $\tilde{O}\big(\Phi_{time}(|\mathcal{U}|, d)\big)$;
- Server memory usage: $\tilde{O}\big(\Phi_{mem}(|\mathcal{U}|, d)\big)$;
- Utility guarantee as follows: for every $\beta' \in (0,1)$ and each $v \in \mathcal{D}$, with probability at least $1-\beta'$, $\mathcal{A}_{oracle}$ returns an estimate $\hat{f}_{\mathcal{U}}[v]$ satisfying $\left|\hat{f}_{\mathcal{U}}[v] - f_{\mathcal{U}}[v]\right| \in O\left((1/\varepsilon) \cdot \sqrt{|\mathcal{U}| \cdot \ln(1/\beta')}\right)$.

For example, when $\mathcal{A}_{oracle}$ is **HRR**, then $\Phi_{time}(|\mathcal{U}|, d) = |\mathcal{U}| + d$ and $\Phi_{mem}(|\mathcal{U}|, d) = d$. Below we state the key result of this section.

Theorem 3.1 (Sketching Framework)**.** *For every $\beta' \in (0,1)$, $\mathcal{A}_{oracle}$ can be converted into a new ε-**LDP** frequency oracle, which has server running time $\tilde{O}(\Phi_{time}(|\mathcal{U}|, \sqrt{|\mathcal{U}|}))$ and memory usage $\tilde{O}(\Phi_{mem}(|\mathcal{U}|, \sqrt{|\mathcal{U}|}))$. Fix some element $v \in \mathcal{D}$ to be given as a query to the new algorithm. With probability at least $1-\beta'$, it returns an estimate $\hat{f}_{\mathcal{U}}[v]$ satisfying*

$$\left|\hat{f}_{\mathcal{U}}[v] - f_{\mathcal{U}}[v]\right| \in O\left((1/\varepsilon) \cdot \sqrt{|\mathcal{U}| \cdot \ln(1/\beta')}\right).$$

In particular, when $\mathcal{A}_{oracle}$ is **HRR**, we obtain a new ε-**LDP** frequency oracle, which we call **HadaOracle**, with the following properties.

Corollary 3.2 (Algorithm **HadaOracle**). *Fix an element $v \in \mathcal{D}$ to be given as a query to **HadaOracle**. With probability at least $1 - \beta'$, **HadaOracle** returns a frequency estimate $\hat{f}_{\mathcal{U}}[v]$ satisfying*

$$\left| \hat{f}_{\mathcal{U}}[v] - f_{\mathcal{U}}[v] \right| \in O\left((1/\varepsilon) \cdot \sqrt{|\mathcal{U}| \cdot \ln(1/\beta')} \right).$$

Each user in $\mathcal{U}$ requires $\tilde{O}(1)$ memory, takes $\tilde{O}(1)$ running time and reports only 1 bit to the server. The server processes the reports in $\tilde{O}(|\mathcal{U}|)$ time and $O(\sqrt{|\mathcal{U}|})$ memory, and answers a query in $\tilde{O}(1)$ time. The $\tilde{O}$ notation hides logarithmic factors in $|\mathcal{U}|$, $|\mathcal{D}|$ and $1/\beta'$.

3.1 A General Framework

To reduce the size of the data domain, we now show how to incorporate *sketching* into **LDP** frequency oracles. Choosing parameters carefully, this leads to significant decrease of the server running time and memory usage without degrading estimation error. The *sketch* we apply is a variant of the Count-Median sketch (Cormode and Yi, 2020), with the framework outlined in Algorithm 1.

Algorithm 1 Sketching Framework

```
Construction
Require: A set of users $\mathcal{U}$; $\varepsilon$-LDP frequency oracle $\mathcal{A}_{oracle}$; element domain $\mathcal{D}$;
1: $k \leftarrow C_K \cdot \ln(4/\beta')$, $m \leftarrow 8e^2 \cdot \sqrt{C_K} \cdot \varepsilon \cdot \sqrt{|\mathcal{U}|}$;
2: Partition $\mathcal{U}$ into $k$ subsets: $\mathcal{U}_1, \ldots, \mathcal{U}_k$.
3: Initialize $k$ pairwise independent hash functions $h_1, \ldots, h_k : \mathcal{D} \to [m]$.
4: for $i \in [k]$ do
5:     The server broadcasts $h_i$ to all users in $\mathcal{U}_i$.
6:     Each user $u \in \mathcal{U}_i$ replaces their element, $v^{(u)} \in \mathcal{D}$, with a new one $h_i(v^{(u)}) \in [m]$.
7:     The server runs an independent copy of $\mathcal{A}_{oracle}$ on $\mathcal{U}_i$, denoted as $\mathcal{A}_{oracle}{}^{(i)}$, for new elements $\{h_i(v^{(u)}) : u \in \mathcal{U}_i\}$.
Query
Require: Element $v \in \mathcal{D}$;
1: for $i \in [k]$ do
2:     Query $\mathcal{A}_{oracle}{}^{(i)}$ for the frequency of $h_i(v)$ over $\{h_i(v^{(u)}) : u \in \mathcal{U}_i\}$, denote the returned estimate as $\hat{f}_{\mathcal{U}_i, h_i}[h_i(v)]$.
3: $\hat{f}_{\mathcal{U}}[v] \leftarrow$ Median $\left( k \cdot \hat{f}_{\mathcal{U}_i, h_i}[h_i(v)] : i \in [k] \right)$.
4: return $\hat{f}_{\mathcal{U}}[v]$
```

Domain Reduction. By mapping the elements from domain $\mathcal{D}$ to $[m]$, we want to reduce the domain size from d to some smaller $m \in \mathbb{N}^+$; we discuss how to set m later. The mapping could be performed via a pairwise independent hash function $h : \mathcal{D} \to [m]$, such that: (1) for each $v \in \mathcal{D}$, it is mapped to $[m]$ uniformly at random; (2) for each pair of distinct $v, v' \in \mathcal{D}$, the probability that they are mapped to the same value is $1/m$. The function h has succinct description of $O(\log d)$ bits (Mitzenmacher and Upfal, 2017). Each user $u \in \mathcal{U}$ is then informed of h and replaces their data $v^{(u)} \in \mathcal{D}$ with the new element $h(v^{(u)}) \in [m]$.

We then invoke **LDP** frequency-oracle $\mathcal{A}_{oracle}$ for estimating frequencies in the new dataset, $\{h(v^{(u)}) : u \in \mathcal{U}\}$. For each $v \in \mathcal{D}$, to obtain an estimate of $f_{\mathcal{U}}[v]$, we return the frequency estimate of $h(v)$ in the new dataset, denoted by $\hat{f}_{\mathcal{U},h}[h(v)]$, provided by $\mathcal{A}_{oracle}$.

Repetition. For $v \in \mathcal{D}$, by the triangle inequality, estimation error $|\hat{f}_{\mathcal{U},h}[h(v)] - f_{\mathcal{U}}[v]|$ is at most:

$$|\hat{f}_{\mathcal{U},h}[h(v)] - f_{\mathcal{U},h}[h(v)]| + |f_{\mathcal{U},h}[h(v)] - f_{\mathcal{U}}[v]|,$$

where $f_{\mathcal{U},h}[h(v)] \doteq |\{v \in \mathcal{U} : h(v^{(u)}) = h(v)\}|$ is the frequency of $h(v)$ in the new dataset. The first term inherits from the estimation error of $\mathcal{A}_{oracle}$; the second term arises from hash collisions of h. The assumption of $\mathcal{A}_{oracle}$ is that the first term is, with probability $1-\beta'$, bounded by $O((1/\varepsilon) \cdot \sqrt{|\mathcal{U}| \cdot \ln(1/\beta')})$.

For the second term, we could set $m \in \tilde{O}(\sqrt{|\mathcal{U}|}/\beta')$. Via Markov's inequality, with probability $1 - \beta'$, it is is $\tilde{O}(\sqrt{|\mathcal{U}|})$. However, when β' is small, e.g., $1/|\mathcal{U}|^c$, for some constant c, this would be unacceptable.

Alternatively, to bound the second term, we could set $m \in \tilde{O}(\sqrt{|\mathcal{U}|})$, bounding with constant probability the second term by $\tilde{O}(\sqrt{|\mathcal{U}|})$. We independently select k pairwise independent hash functions $h_1, \ldots, h_k$. To run the standard Count-Median Sketch, we would apply each $h_i, i \in [k]$ to each of the users, and for each $i \in [k]$ invoke $\mathcal{A}_{oracle}$ independently to estimate the elements' frequencies over $\{h_i(v^{(u)}) : u \in \mathcal{U}\}$. For $v \in \mathcal{D}$, the median over $i \in [k]$ of $\hat{f}_{\mathcal{U},h_i}[h_i(v)]$ is returned as its frequency estimate.

But this scheme requires each user to participate in k frequency oracles, which would degrade privacy according to the Composition Theorem (Dwork and Roth, 2014, Theorem 3.14). This motivates partitioning $\mathcal{U}$ into k subsets, denoted as $\mathcal{U}_1, \ldots, \mathcal{U}_k$. For each $i \in [k]$, the users in subset $\mathcal{U}_i$ map their elements with hash function h_i, and an independent copy of $\mathcal{A}_{oracle}$ is applied to $\mathcal{U}_i$, to estimate the frequencies of $\{h_i(v^{(u)}) : u \in \mathcal{U}_i\}$.

Here we study two partitioning schemes.

- *Independent partitioning.* Here, each user is put into one of the subsets $\{\mathcal{U}_i : i \in [k]\}$ uniformly and independently at random.

- *Permutation partitioning.* Here we randomly permute $\mathcal{U}$: the first $|\mathcal{U}|/k$ users in the permutation become $\mathcal{U}_1$, the next $|\mathcal{U}|/k$ become $\mathcal{U}_2$, etc.

Compared to *independent partitioning*, permutation partitioning creates subsets of equal size. Since each user participates in only one copy of $\mathcal{A}_{oracle}$, Algorithm 1 is ε-**LDP**. It remains to prove the utility guarantee of Algorithm 1.

3.2 Utility Analysis

Two Kinds of Frequencies. First, consider the frequencies of elements before hashing in each subset. For each $i \in [k]$, and $v \in \mathcal{D}$, define $f_{\mathcal{U}_i}[v] \doteq |\{u \in \mathcal{U}_i : v^{(u)} = v\}|$ to be the frequency of v in the set $\{v^{(u)} : u \in \mathcal{U}_i\}$. It is a random variable, whose randomness inherits from the partitioning. But for each partitioning scheme, $\mathbb{E}\left[f_{\mathcal{U}_i}[v]\right] = f_{\mathcal{U}}[v]/k$.

Second, consider the frequencies of the hashed elements in each subset. For each $i \in [k]$ and $w \in [m]$, let $f_{\mathcal{U}_i,h_i}[w] \doteq |\{u \in \mathcal{U}_i : h_i(v^{(u)}) = w\}|$ be the number of users in $\mathcal{U}_i$ whose item are hashed to w. It is also a random variable, whose randomness arises from the partitioning, and from the hash function h_i. It holds that $\mathbb{E}[f_{\mathcal{U}_i,h_i}[w]] = |\mathcal{U}_i|/m$.

Three Kinds of Errors. For each $i \in [k]$ and $v \in \mathcal{D}$, let $\hat{f}_{\mathcal{U}_i,h_i}[h_i(v)]$ be the estimate of $f_{\mathcal{U}_i,h_i}[h_i(v)]$ by $\mathcal{A}_{oracle}$. We are interested in the deviation of $k \cdot \hat{f}_{\mathcal{U}_i,h_i}[h_i(v)]$ from $f_{\mathcal{U}}[v]$, which can be decomposed into three parts:

1. $\lambda_1(i,v) \doteq k f_{\mathcal{U}i}[v] - f_{\mathcal{U}}[v]$.
2. $\lambda_2(i,v) \doteq k f_{\mathcal{U}_i,h_i}[h_i(v)] - k f_{\mathcal{U}i}[v]$.
3. $\lambda_3(i,v) \doteq k \hat{f}_{\mathcal{U}_i,h_i}[h_i(v)] - k f_{\mathcal{U}_i,h_i}[h_i(v)]$.

Define $\mathit{Err}(i,v) \doteq k\hat{f}_{\mathcal{U}_i,h_i}[h_i(v)] - f_{\mathcal{U}}[v]$. Its absolute value is bounded by a triangle inequality

$$|\mathit{Err}(i,v)| \le |\lambda_1(i,v)| + |\lambda_2(i,v)| + |\lambda_3(i,v)| . \qquad (1)$$

Four Kinds of Good Sets. We are interested in the following four kinds of *good sets*, that play important roles in bounding the estimation error of Algorithm 1. First, define

$$\mathbb{G}d\mathbb{S}et_0 \doteq \{i \in [k] : k|\mathcal{U}_i| \in \Theta(|\mathcal{U}|)\} .$$

Then, for each $v \in \mathcal{D}$, define

$$\mathbb{G}d\mathbb{S}et_1(v) \doteq \Big\{ i \in [k] : |\lambda_1(i,v)| \in O\Big(\sqrt{|\mathcal{U}| \ln \frac{1}{\beta'}}\Big)\Big\} .$$

Finally, for $j = 2, 3$, define

$$\mathbb{G}d\mathbb{S}et_j(v) \doteq \Big\{ i \in [k] : |\lambda_j(i,v)| \in O\Big(\frac{1}{\varepsilon}\sqrt{k|\mathcal{U}_i| \ln \frac{1}{\beta'}}\Big)\Big\} .$$

The following result is the key to the utility guarantee.

Theorem 3.3. *With probability at least $1 - \beta'/4$, it holds that $|\mathbb{G}d\mathbb{S}et_0| > (1 - 1/8)k$. And for each $v \in \mathcal{D}$ and each $j \in [3]$, with probability at least $1 - \beta'/4$, it holds that $|\mathbb{G}d\mathbb{S}et_j(v)| > (1 - 1/8)k$.*

Theorem 3.3 holds for both independent partitioning and permutation partitioning. The proof is technical, so we defer to the end of this subsection. For now, we prove the utility guarantee of Algorithm 1.

Corollary 3.4. *For each $v \in \mathcal{D}$, with probability at least $1 - \beta'$, the $\hat{f}_{\mathcal{U}}[v]$ returned by Algorithm 1 satisfies*

$$\left|\hat{f}_{\mathcal{U}}[v] - f_{\mathcal{U}}[v]\right| \in O\left((1/\varepsilon) \cdot \sqrt{|\mathcal{U}| \cdot \ln(1/\beta')}\right) .$$

Proof of Corollary 3.4. By Theorem 3.3, and a union bound, with probability at least $1 - \beta'$, we have

$$\left|\big(\cap_{j\in[3]} \mathbb{G}d\mathbb{S}et_j(v)\big) \cap \mathbb{G}d\mathbb{S}et_0\right| > k/2 . \qquad (2)$$

For each $i \in \big(\cap_{j\in[3]} \mathbb{G}d\mathbb{S}et_j(v)\big) \cap \mathbb{G}d\mathbb{S}et_0$, since $k|\mathcal{U}_i| \in \Theta(|\mathcal{U}|)$, for $j = 2$ or 3, it holds that

$$|\lambda_j(i,v)| \in O\Big(\frac{1}{\varepsilon}\sqrt{k|\mathcal{U}_i| \ln \frac{1}{\beta'}}\Big) \subset O\Big(\frac{1}{\varepsilon}\sqrt{|\mathcal{U}| \ln \frac{1}{\beta'}}\Big).$$

Therefore, via Inequality (1), it holds that

$$|\mathit{Err}(i,v)| \in O\Big((1/\varepsilon) \cdot \sqrt{|\mathcal{U}| \ln(1/\beta')}\Big) .$$

We finish by combining this error bound with $\hat{f}_{\mathcal{U}}[v] = \mathbf{Median}_{i\in[k]}\, k \cdot \hat{f}_{\mathcal{U}_i,h_i}[h(v)]$ and inequality (2). □

Proof Outline for Theorem 3.3. This is a sketch of a full proof that appears in the Appendix. As they are easier, we first bound the sizes of $\mathbb{G}d\mathbb{S}et_2(v)$ and $\mathbb{G}d\mathbb{S}et_3(v)$.

Bounding the Size of $\mathbb{G}d\mathbb{S}et_2(v)$. Observe that for each $i \in [k]$, the (scaled) errors $|\lambda_2(i,v)|/k = |f_{\mathcal{U}_i,h_i}[h(v)] - f_{\mathcal{U}i}[v]|$ result from hash collisions. Since h_i is pairwise independent, the expected size of collision is at most $|\mathcal{U}_i|/m$, i.e., $\mathbb{E}\left[|\lambda_2(i,v)|/k\right] \le |\mathcal{U}_i|/m$. Recall that Algorithm 1 initializes $k = C_K \cdot \ln(4/\beta')$ and $m = 8e^2 \cdot \sqrt{C_K} \cdot \varepsilon \cdot \sqrt{|\mathcal{U}|}$ for some constant C_K. Via Markov's inequality, and that $|\mathcal{U}| \ge |\mathcal{U}_j|$, we have

$$\Pr\left[|\lambda_2(i,v)| \ge (1/\varepsilon)\sqrt{k|\mathcal{U}_i| \ln(4/\beta')}\right] \le 1/(8e^2) .$$

Therefore, $\Pr[i \ne \mathbb{G}d\mathbb{S}et_2(v)]$ is upper bounded by $1/(8e^2)$. As the $h_1, \ldots, h_k$ are chosen independently, the indicators of not being in $\mathbb{G}d\mathbb{S}et_2(v)$ are independent over $i \in [k]$. By a Chernoff bound, we can prove that, if $k = C_K \cdot \ln(4/\beta')$ for some large enough constant C_K, then

$$\Pr[|\{i \in [k] : i \ne \mathbb{G}d\mathbb{S}et_2(v)\}| \ge k/8] \le \beta'/4 .$$

Bounding the Size of $\mathbb{G}d\mathbb{S}et_3(v)$. Via the assumption of $\mathcal{A}_{oracle}$, for $i \in [k]$, with probability at most $1/(8e^2)$,

$$|\hat{f}_{\mathcal{U}_i,h_i}[h_i(v)] - f_{\mathcal{U}_i,h_i}[h_i(v)]| \notin O\left(\frac{1}{\varepsilon}\sqrt{|\mathcal{U}_i| \cdot \ln(8e^2)}\right).$$

Scaling both sides by a factor of k, we get $|\lambda_3(i,v)| \notin O((1/\varepsilon)\sqrt{k^2|\mathcal{U}_j|})$. Replacing one factor k with $C_K \cdot \ln(4/\beta')$, we have $|\lambda_3(i,v)| \notin O((1/\varepsilon)\sqrt{k|\mathcal{U}_j| \cdot \ln(1/\beta')})$. Since the indicators of being in $\mathbb{G}d\mathbb{S}et_3(v)$ are independent, there exists some constant C_K, s.t.,

$$\Pr[|\{i \in [k] : i \neq \mathbb{G}d\mathbb{S}et_3(v)\}| \geq k/8] \leq \beta'/4\,.$$

Bounding the Size of $\mathbb{G}d\mathbb{S}et_0$. For permutation partition, it holds that $|\mathcal{U}_i| = |\mathcal{U}|/k$, $\forall i \in [k]$. Therefore, $|\mathbb{G}d\mathbb{S}et_0(v)| = k$. For independent partitioning, analyzing $\mathbb{G}d\mathbb{S}et_0$ is not trivial, as the $|\mathcal{U}_i|$ are not independent. Therefore, we consider the deviations of the subset sizes as a whole: define $\Delta_0 \doteq \sum_{i\in[k]} \big||\mathcal{U}_i| - |\mathcal{U}|/k\big|$, which measures the distance between vector $(|\mathcal{U}_1|, \ldots, |\mathcal{U}_k|) \in \mathbb{R}^k$ and its expectation. Via the McDiarmid inequality (Mitzenmacher and Upfal, 2017), we prove that, there exists some constant C_0, s.t., for every choice of positive integer k, with probability at least $1 - \beta'/4$: $\Delta_0 \leq C_0\sqrt{|\mathcal{U}|\ln(4/\beta')}$.

It follows that $\sum_{i\in[k]} \big|k|\mathcal{U}_i| - |\mathcal{U}|\big| \leq kC_0\sqrt{|\mathcal{U}|\ln(4/\beta')}$. By a Markov inequality-like argument, the number of $i \in [k]$, such that $\big|k|\mathcal{U}_i| - |\mathcal{U}|\big| \geq 8C_0\sqrt{|\mathcal{U}|\ln(4/\beta')}$ is bounded by $(1/8)k$, which finishes the proof.

Bounding the Size of $\mathbb{G}d\mathbb{S}et_1(v)$. Similarly, we consider the deviations as a whole, and define $\Delta_1 \doteq \sum_{i\in[k]} \|f_{\mathcal{U}i} - f_{\mathcal{U}}/k\|_2$, where $\|f_{\mathcal{U}i} - f_{\mathcal{U}}/k\|_2 \doteq \sqrt{\sum_{v'\in\mathcal{D}}(f_{\mathcal{U}i}[v'] - f_{\mathcal{U}}[v']/k)^2}$. Note that $\forall i \in [k]$, $|\lambda_1(i,v)| \leq k\,\|f_{\mathcal{U}i} - f_{\mathcal{U}}/k\|_2$.

Via the martingale concentration inequalities (the McDiarmid inequality and the Azuma–Hoeffding inequality (Mitzenmacher and Upfal, 2017; Chung and Lu, 2006)), we prove for both independent partitioning and permutation, that there exists some constant C_1, s.t., for every choice of positive integer k, with probability at least $1 - \beta'/4$: $\Delta_1 \leq C_1\sqrt{|\mathcal{U}|\ln(4/\beta')}$.

Hence, $k\sum_{i\in[k]} \|f_{\mathcal{U}i} - f_{\mathcal{U}}/k\|_2 \leq kC_1\sqrt{|\mathcal{U}|\ln(4/\beta')}$. By a counting argument, the number of $i \in [k]$, such that $k\,\|f_{\mathcal{U}i} - f_{\mathcal{U}}/k\|_2 \geq 8C_1\sqrt{|\mathcal{U}|\ln(4/\beta')}$ is bounded by $(1/8)k$, which finishes the proof. □

3.3 Comparison With Previous Approaches

The seminal work of Bassily et al. (2017) was the first to provide rigorous analysis for ε-**LDP** frequency oracles with sketching methods. We differ from their approach thus: (1) They apply Count-Sketch instead of Count-Median Sketch. (2) Similar to our bounding $|\mathbb{G}d\mathbb{S}et_0|$, they invoke a technique called *Poisson Approximation* (Mitzenmacher and Upfal, 2017), which approximates the distribution of $|\mathcal{U}_i|, i \in [k]$ by a set of k independent Poisson random variables. This results in a setting of $k \in \Theta(\ln(|\mathcal{U}|/\beta'))$, and subsequently a sub-optimal utility guarantee of $O\big((1/\varepsilon)\cdot\sqrt{|\mathcal{U}|\cdot\ln(|\mathcal{U}|/\beta')}\big)$. (3) Finally, though invoking **HRR** as a sub-routine, their work does not exploit the fast Hadamard transform. Hence their oracle answers a frequency query in $\tilde{O}(\sqrt{|\mathcal{U}|})$ time, instead of $\tilde{O}(1)$.

The recent experimental study by Cormode et al. (2021) provides inspiring insights. They propose to view existing algorithms as different combinations of *sketching methods* (for domain reduction) and frequency oracle algorithms (constructed over the reduced domain). Their work empirically evaluates performance of the sketches based on different values of k and m, without corresponding theoretical analysis.

Lastly, we discuss briefly the Count-Median Sketch applied by our algorithm. Compared to the standard Count-Median Sketch, which applies each of the k hash functions to the data of all users, our version partitions the users into k subsets, and applies each hash function only to a particular subset. The primary reason for doing so in our work is to protect the privacy of the users. But this technique can also be applied to applications where there is no requirement for privacy protection. It reduces the time for processing a user's report from $O(k)$ to $O(1)$. Based on our technique for analysing the size of $\mathbb{G}d\mathbb{S}et_1$, this introduces an additional error of $O(\sqrt{|\mathcal{U}|\ln(1/\beta')})$ to an element's frequency estimate. In cases where the allowed error is $\Omega(\sqrt{|\mathcal{U}|\ln(1/\beta')})$, this is practical.

4 Succinct Histogram

We show how to construct a succinct histogram efficiently, based on the **HadaOracle** discussed in previous section. Our new algorithm, **HadaHeavy** improves the **TreeHist** algorithm by Bassily et al. (2017).

Theorem 4.1. *Let $\mathcal{U}$ be a set of n users each holding an element from some finite domain $\mathcal{D}$ of size d, $\varepsilon \in (0,1)$ be the privacy parameter and $\beta \in (0,1)$ be the specified failure probability.* ***HadaHeavy*** *is an ε-**LDP** algorithm, based on hierarchical search method, that returns an **S-Hist** set of (element, estimate) pairs of size $\tilde{O}(\sqrt{n})$, where*

$$\lambda \in O((1/\varepsilon)\cdot\sqrt{n\cdot(\ln d)\cdot(1+(\ln(1/\beta)\ /\ln n))})\,.$$

Each user in $\mathcal{U}$ requires $\tilde{O}(1)$ memory, takes $\tilde{O}(1)$ running time and reports only 1 bit to the server. The *server processes the reports in $\tilde{O}(n)$ time and $\tilde{O}(\sqrt{n})$ memory. The $\tilde{O}$ notation hides logarithmic factors in n, d and $1/\beta$.*

4.1 HadaHeavy

HadaHeavy represents element in $\mathcal{D}$ with an alphabet of size $\sqrt{n}$.

Base-$\sqrt{n}$ representation. Let $\Lambda \doteq \{0, 1, \ldots, \sqrt{n}-1\}$ be an alphabet of size $\sqrt{n}$ (for simplicity, we assume that $\sqrt{n}$ is an integer), and $L \doteq 2 \cdot (\log d)/\log n$. Each element in $\mathcal{D}$ can be encoded as a unique string in $\Lambda^L = \{0, 1, \ldots, \sqrt{n}-1\}^L$.

Prefix. For each $v \in \mathcal{D}, \tau \in [L]$, let $v[1 : \tau]$ be the first τ characters in v's base-$\sqrt{n}$ representation, which is called a *prefix* of v. Let $\Lambda^0 \doteq \{\perp\}$ be the set consisting of the empty string. For each $\tau \in [L]$, Λ^τ the set of all possible strings of length τ. Further, for each string $\boldsymbol{s} \in \Lambda^\tau$, define the frequency of $\boldsymbol{s}$ to be $f_{\mathcal{U}}[\boldsymbol{s}] \doteq |\{u \in \mathcal{U} : v^{(u)}[1 : \tau] = \boldsymbol{s}\}|$.

Child Set. For each $0 \le \tau < L$, and each string $\boldsymbol{s} \in \Lambda^\tau$, the child set of $\boldsymbol{s}$, denoted as $\boldsymbol{s} \times \Lambda$, is defined as $\boldsymbol{s} \times \Lambda \doteq \{\boldsymbol{s} \circ \boldsymbol{t} : \boldsymbol{t} \in \Lambda\} \subset \Lambda^{\tau+1}$, where $\boldsymbol{s} \circ \boldsymbol{t}$ is the concatenation of the strings $\boldsymbol{s}$ and $\boldsymbol{t}$.

The key motivation for the *hierarchical searching method* is that, if an element $v \in \mathcal{D}$ is frequent, so is each of its prefixes.

Overview of HadaHeavy. The goal of the algorithm is to identify a set of elements in $\mathcal{D}$, called heavy hitters, whose frequencies are no less than some threshold, λ (to be determined later). Clearly, for each $\tau \in [L]$,

$$f_{\mathcal{U}}[v[1 : \tau]] \ge f_{\mathcal{U}}[v] \ge \lambda\,.$$

Assuming that we know the exact values of frequencies of the strings. We can search for heavy hitters as follows. First, we initialize a sequence of empty sets $\mathcal{P}_1, \ldots, \mathcal{P}_L$, which we call *search sets*. Then, we examine all strings in Λ. If $\boldsymbol{s} \in \Lambda$ has frequency $f_{\mathcal{U}}[\boldsymbol{s}] \ge \lambda$, then we put it into $\mathcal{P}_1$. Next, for each string $\boldsymbol{s} \in \mathcal{P}_1$, we check each string $\boldsymbol{s}'$ from its child set $\boldsymbol{s} \times \Lambda$. If $\boldsymbol{s}'$ has frequency $f_{\mathcal{U}}[\boldsymbol{s}'] \ge \lambda$, then we put it into $\mathcal{P}_2$. In general, for $\tau \ge 2$, we can construct $\mathcal{P}_\tau$ after $\mathcal{P}_{\tau-1}$ is constructed. Finally $\mathcal{P}_L$ should contain all heavy hitters in $\mathcal{D}$.

Frequency Oracles. As the exact values of the frequencies of the strings are not available, we want to learn their estimates. For each $\tau \in [L]$, we construct a frequency oracle (**HadaOracle**) to estimate the frequencies of the strings in Λ^τ. We want to avoid each user participating in every frequency oracle: a user reporting L times to the server would degenerate the privacy guarantee of the algorithm. Therefore, we partition the set of users $\mathcal{U}$ into L subsets $\mathbb{U}_1, \mathbb{U}_2, \ldots, \mathbb{U}_L$. As before, this is performed by either *independent partitioning* or *permutation partitioning*, introduced in Section 3.1. As $L \in O(\log d)$, by Corollary 3.2, the server uses in total $\tilde{O}(n)$ processing time and $\tilde{O}(\sqrt{n})$ processing memory to construct these oracles.

For each $\tau \in [L]$ and each $\boldsymbol{s} \in \Lambda^\tau$, let $f_{\mathbb{U}_\tau}[\boldsymbol{s}] \doteq |\{u \in \mathbb{U}_\tau : v^{(u)}[1 : \tau] = \boldsymbol{s}\}|$ be the frequency of $\boldsymbol{s}$ in $\mathbb{U}_\tau$, and $\hat{f}_{\mathbb{U}_\tau}[\boldsymbol{s}]$ be its estimate by **HadaOracle**. As $\mathbb{E}[L \cdot f_{\mathbb{U}_\tau}[\boldsymbol{s}]] = f_{\mathcal{U}}[\boldsymbol{s}]$, we use $\hat{f}_{\mathcal{U}}[\boldsymbol{s}] \doteq L \cdot \hat{f}_{\mathbb{U}_\tau}[\boldsymbol{s}]$ as an estimate of $f_{\mathcal{U}}[\boldsymbol{s}]$: its estimation error is established by the following theorem, proven in the Appendix.

Theorem 4.2. *For each $\tau \in [L]$, fix some query string $\boldsymbol{s} \in \Lambda^\tau$ for the frequency estimate. It holds that, with probability $1 - \beta'$,*

$$|\hat{f}_{\mathcal{U}}[\boldsymbol{s}] - f_{\mathcal{U}}[\boldsymbol{s}]| \in O((1/\varepsilon)\sqrt{n \cdot (\log d) \cdot (\ln(1/\beta'))/\ln n})\,.$$

There are two sources of error. First, for each $\tau \in [L]$, the frequency distribution in $\mathbb{U}_\tau$ deviates from its expectation. We bound the ℓ_2 distance between the distribution in $\mathbb{U}_\tau$ and its expectation by martingale methods. The second kind of error inherits from **HadaOracle**.

Modified Search Strategy. Only having access to estimates of the frequencies of the prefixes, we need to modify the criterion for adding elements to $\mathcal{P}_\tau$.

Let $\lambda' \doteq (C_\lambda/\varepsilon)\sqrt{n \cdot (\log d) \cdot (\ln(n/\beta))/\ln n}$, where C_λ is some constant we will determine later. Define $\mathcal{P}_0 \doteq \{\perp\}$. The $\mathcal{P}_\tau$ are iteratively constructed according to the following criterion:

$$\mathcal{P}_\tau \leftarrow \{\boldsymbol{s} \in \mathcal{P}_{\tau-1} \times \Lambda : \hat{f}_{\mathcal{U}}[\boldsymbol{s}] \ge 2\lambda'\},\ \forall \tau \in [L]\,,$$

where $\mathcal{P}_{\tau-1} \times \Lambda \doteq \{\boldsymbol{s} = \boldsymbol{s}_1 \circ \boldsymbol{s}_2 : \boldsymbol{s}_1 \in \mathcal{P}_{\tau-1}, \boldsymbol{s}_2 \in \Lambda\}$. Finally, after $\mathcal{P}_L$ is constructed, its elements are returned as heavy hitters.

4.2 Analysis

Since each user participates in only one frequency oracle, the algorithm is ε-**LDP**. It remains to analyze the utility guarantee, running time and memory usage of the *modified search strategy.*

Theorem 4.3. *Let $\lambda \doteq 3 \cdot \lambda'$. With probability at least $1 - \beta$, it is guaranteed that for each $\tau \in [L]$, and each $\boldsymbol{s} \in \Lambda^\tau$: (1) if $f_{\mathcal{U}}[\boldsymbol{s}] \ge \lambda$, then $\boldsymbol{s} \in \mathcal{P}_\tau$; (2) and for each $\boldsymbol{s} \in \mathcal{P}_\tau$, the frequency estimate $\hat{f}_{\mathcal{U}}[\boldsymbol{s}]$ satisfies $|\hat{f}_{\mathcal{U}}[\boldsymbol{s}] - f_{\mathcal{U}}[\boldsymbol{s}]| \le \lambda'$. Constructing the $\mathcal{P}_\tau$ for all $\tau \in [L]$ has $\tilde{O}(n)$ running time and $\tilde{O}(\sqrt{n})$ memory usage.*

We sketch the proof here, details are in the Appendix.

Proof Outline for Theorem 4.3. We focus on the estimation errors of prefixes from a fixed set.

Definition 4.4 (Candidate Set)**.** Define $\Gamma_0 \doteq \{\bot\}$ to be the set of the empty string, and for $\tau \in [L]$, $\Gamma_\tau \doteq \{\boldsymbol{s} \in \Lambda^\tau : f_{\mathcal{U}}[\boldsymbol{s}] \geq \lambda'\}$, the set of prefixes of length τ whose frequency is at least λ'. For $\tau < L$, the child set of Γ_τ is defined as $\Gamma_\tau \times \Lambda \doteq \{\boldsymbol{s} = \boldsymbol{s}_1 \circ \boldsymbol{s}_2 : \boldsymbol{s}_1 \in \Gamma_\tau, \boldsymbol{s}_2 \in \Lambda\}$, where $\boldsymbol{s}_1 \circ \boldsymbol{s}_2$ is the concatenation of $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$. The *candidate* set is defined as $\Gamma \doteq \cup_{0 \leq \tau < L} (\Gamma_\tau \times \Lambda)$.

Note that for each $\tau \in [L]$, we have $|\Gamma_\tau| \leq n/\lambda' \leq \sqrt{n}$. Hence, $|\Gamma| = \sum_{0 \leq \tau < L} |\Gamma_\tau \times \Lambda| \leq L\sqrt{n} \cdot \sqrt{n} \in \tilde{O}(n)$. By applying Theorem 4.2 with $\beta' = \beta/(nL)$ and the union bound over all $\boldsymbol{s} \in \Gamma$, we have:

Corollary 4.5. *There exists some constant C_λ, such that with probability at least $1 - \beta$, it holds that $\max_{\boldsymbol{s} \in \Gamma} |\hat{f}_{\mathcal{U}}[\boldsymbol{s}] - f_{\mathcal{U}}[\boldsymbol{s}]| \leq \lambda'$, where*

$$\lambda' = (C_\lambda/\varepsilon)\sqrt{n \cdot (\log d) \cdot (\ln(n/\beta))/\ln n} \, .$$

Conditioned all strings in Γ having estimation error λ', we can prove by induction that the *modified search strategy* only inspects the frequencies of the strings from Γ, in order to construct the $\mathcal{P}_\tau, \tau \in [L]$. Therefore, the strings added to $\mathcal{P}_\tau, \tau \in [L]$ have estimation errors bounded by λ'. Moreover, for each $\tau \in [L]$, and each $\boldsymbol{s} \in \Lambda^\tau$, if $f_{\mathcal{U}}[\boldsymbol{s}] \geq \lambda = 3\lambda'$, then $\boldsymbol{s} \in \Gamma$, and we can prove that $\boldsymbol{s}$ will be added to $\mathcal{P}_\tau$. The details are included in the of the complete proof in Appendix.

To analyze the running time and memory usage, observe that the *modified search strategy* invokes L frequency oracles. By Theorem 3.2, they have total construction time $\tilde{O}(n)$ and memory usage $\tilde{O}(\sqrt{n})$. Since each frequency query takes $\tilde{O}(1)$ time, and all strings queried belong to Γ, the total query time is bounded by $|\Gamma| \in \tilde{O}(n)$, which finishes the proof. □

4.3 Comparison With Previous Approaches

Among the previous algorithms that identify heavy hitters based on hierarchical search, **TreeHist** (Bassily et al., 2017) provides the best known error guarantee of $O((1/\varepsilon) \cdot \sqrt{n \cdot (\ln d) \cdot \ln(n/\beta)})$. Our algorithm reduces this error to $O((1/\varepsilon) \cdot \sqrt{n \cdot (\ln d) \cdot (1 + (\ln(1/\beta) \ / \ln n))})$. There are two major differences between our algorithm and **TreeHist**. First, the algorithm **TreeHist** considers base-2 representation of elements in $\mathcal{D}$, instead of base-$\sqrt{n}$ representation. Each element in $\mathcal{D}$ is encoded as a binary string of length $\log d$. This requires the algorithm to partition the user set $\mathcal{U}$ into $\log d$ subsets, which results in smaller subset sizes and larger estimation error than our algorithm. Second, the frequency oracle used by **TreeHist** does not exploit the fast Hadamard transform. Its frequency oracle answers a frequency query in $\tilde{O}(\sqrt{n})$ time. In comparison, **HadaOracle** answers a query in $\tilde{O}(1)$ time. Recent works (Wang et al., 2021; Cormode et al., 2021) observe that, instead of identifying prefixes of the heavy hitters with increasing lengths, one character at a time, we can identify such prefixes by several characters at each step. This reduces the number of steps required to reach the full length strings. Indeed, using a large alphabet to represent the elements in $\mathcal{D}$ (e.g., an alphabet of size $\sqrt{n}$, as we proposed) achieves the same effect. This strategy is effective empirically (Wang et al., 2021; Cormode et al., 2021), but there are no theoretical guarantees. We believe our work improves understanding of these high-quality experimental results.

5 Related Work

Frequency Oracle. We briefly document the development of frequency oracles in recent years. Bassily and Smith (2015) described a frequency oracle that achieves error $O((1/\varepsilon) \cdot \sqrt{n \log(1/\beta)})$. However, it needs $\tilde{O}(n^2)$ random bits to describe a random matrix, and answers a query in $O(n)$ time. In 2017, a similar version with simplified analysis, called **ExplicitHist**, was studied by Bassily et al. (2017). **ExplicitHist** achieves the same estimation error, but requires only $\tilde{O}(1)$ random bits to describe the random matrix. It answers a query in $O(n)$ time. Other optimizations have been proposed. First documented in (Nguyên et al., 2016), and widely used in **LDP** literature (Bassily et al., 2017; Apple, 2017; Cormode et al., 2019), the **HRR** algorithm uses the Hadamard matrix to replace the random matrix without increasing the estimation error. The matrix does not need to be generated explicitly and each of its entries can be computed in $\tilde{O}(1)$ time when needed. The **HRR** answers a query in $O(\min\{n, d\})$ time, or with pre-processing time $\tilde{O}(d)$, answers each query in $\tilde{O}(1)$ time. In 2019, Acharya et al. proposed a variant of **HRR** which does not rely on public randomness. The protocol shares similar performance guarantees to the original, but can be modified to support frequency estimation in low privacy regime ($\varepsilon > 1$) (Ghazi et al., 2021). Finally, Bassily et al. (2017) also applied Count-Sketch (Charikar et al., 2002) to reduce the domain size of the elements. Their frequency oracles, **FreqOracle** and **Hashtogram**, have server running time $\tilde{O}(n)$ and memory usage $\tilde{O}(\sqrt{n})$. But these algorithms have sub-optimal estimation error of $O((1/\varepsilon) \cdot \sqrt{n \log(n/\beta)})$.

Succinct Histogram. For the succinct histogram problem, Bassily and Smith (2015) proposed the first polynomial-time algorithm that has worst-case error $O((\log^{1.5}(1/\beta)) \cdot (1/\varepsilon) \cdot \sqrt{n \log d})$. However, it has server running time $\tilde{O}(n^{2.5})$ and user time $\tilde{O}(n^{1.5})$, which is not practical. Bassily et al. (2017) proposed two improved algorithms, **TreeHist** and **Bitstogram**, which involve different techniques. **TreeHist** searches for the heavy hitters via a prefix tree;

Bitstogram hashes elements into a smaller domain and identifies a noisy version of the heavy hitters. The recovery of the true heavy hitters relies on error-correcting codes. The former algorithm has error $O((1/\varepsilon)\cdot\sqrt{n\cdot(\log d)\cdot\log(n/\beta)})$, while the latter has error $O((1/\varepsilon)\cdot\sqrt{n\cdot(\log(d/\beta))\cdot\log(1/\beta)})$; each achieves almost-optimal error, but **TreeHist** is inferior to **Bitstogram** by a factor of $\sqrt{\log n}$. Importantly, each algorithm has server time $\widetilde{O}(n)$ and user time $\widetilde{O}(1)$.

Due to the sophistication of error-correcting codes, of the two algorithms presented in (Bassily et al., 2017), only **TreeHist** was implemented and experimented. Bun et al. (2019) further refined **Bitstogram** based on the list-recoverable code, which involves identifying spectral clusters in a derived graph. Their new algorithm, **PrivateExpanderSketch** (Bun et al., 2019), achieved an optimal error of $O((1/\varepsilon)\cdot\sqrt{n\cdot\log(d/\beta)})$. Again, this style of algorithm has yet to be implemented.

We observe finally that *Sketching methods* and *hierarchical searching methods* are not only applied to the **LDP** model, but also to other models of DP for frequency estimation, for example the shuffle model (Luo et al., 2021; Balle et al., 2019; Ghazi et al., 2021).

Acknowledgements

This research is supported by an Australian Government Research Training Program (RTP) Scholarship.

References

R. Bassily, K. Nissim, U. Stemmer, and A. G. Thakurta, "Practical locally private heavy hitters," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds., 2017, pp. 2288–2296.

Ú. Erlingsson, V. Pihur, and A. Korolova, "RAPPOR: randomized aggregatable privacy-preserving ordinal response," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014*, G. Ahn, M. Yung, and N. Li, Eds. ACM, 2014, pp. 1054–1067.

Apple, "Learning with privacy at scale," *Apple Machine Learning Journal*, vol. 2017, no. 1, pp. 1–25, 2017.

G. C. Fanti, V. Pihur, and Ú. Erlingsson, "Building a RAPPOR with the unknown: Privacy-preserving learning of associations and data dictionaries," *Proc. Priv. Enhancing Technol.*, vol. 2016, no. 3, pp. 41–61, 2016.

P. Voigt and A. Von dem Bussche, "The EU general data protection regulation (GDPR)," *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, vol. 10, p. 3152676, 2017.

J. Tang, A. Korolova, X. Bai, X. Wang, and X. Wang, "Privacy loss in Apple's implementation of differential privacy on MacOS 10.12," *CoRR*, vol. abs/1709.02753, 2017.

B. Ding, J. Kulkarni, and S. Yekhanin, "Collecting telemetry data privately," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds., 2017, pp. 3571–3580.

C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Found. Trends Theor. Comput. Sci.*, vol. 9, no. 3-4, pp. 211–407, 2014.

R. Bassily and A. D. Smith, "Local, private, efficient protocols for succinct histograms," in *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, R. A. Servedio and R. Rubinfeld, Eds. ACM, 2015, pp. 127–135.

M. Bun, J. Nelson, and U. Stemmer, "Heavy hitters and the structure of local privacy," *ACM Trans. Algorithms*, vol. 15, no. 4, pp. 51:1–51:40, 2019.

G. Cormode, S. Maddock, and C. Maple, "Frequency estimation under local differential privacy [experiments, analysis and benchmarks]," *CoRR*, vol. abs/2103.16640, 2021.

S. L. Warner, "Randomized response: A survey technique for eliminating evasive answer bias," *Journal of the American Statistical Association*, vol. 60, no. 309, pp. 63–69, 1965.

T. Wang, J. Blocki, N. Li, and S. Jha, "Locally differentially private protocols for frequency estimation," in *26th USENIX Security Symposium, USENIX Security 2017, Vancouver, BC, Canada, August 16-18, 2017*, E. Kirda and T. Ristenpart, Eds. USENIX Association, 2017, pp. 729–745.

R. Bassily, K. Nissim, U. Stemmer, and A. Thakurta, "Practical locally private heavy hitters," *J. Mach. Learn. Res.*, vol. 21, pp. 16:1–16:42, 2020.

N. Wang, X. Xiao, Y. Yang, T. D. Hoang, H. Shin, J. Shin, and G. Yu, "Privtrie: Effective frequent term discovery under local differential privacy," in *34th IEEE International Conference on Data Engineering, ICDE 2018, Paris, France, April 16-19, 2018.* IEEE Computer Society, 2018, pp. 821–832.

T. T. Nguyên, X. Xiao, Y. Yang, S. C. Hui, H. Shin, and J. Shin, "Collecting and analyzing data from smart device users with local differential privacy," *CoRR*, vol. abs/1606.05053, 2016.

G. Cormode, T. Kulkarni, and D. Srivastava, "Answering range queries under local differential privacy," *Proc. VLDB Endow.*, vol. 12, no. 10, pp. 1126–1138, 2019.

G. Cormode and K. Yi, *Small Summaries for Big Data*. Cambridge University Press, 2020.

M. Mitzenmacher and E. Upfal, *Probability and computing: Randomization and probabilistic techniques in algorithms and data analysis*. Cambridge University Press, 2017.

F. Chung and L. Lu, "Concentration inequalities and martingale inequalities: a survey," *Internet Math.*, vol. 3, no. 1, pp. 79–127, 2006.

T. Wang, N. Li, and S. Jha, "Locally differentially private heavy hitter identification," *IEEE Trans. Dependable Secur. Comput.*, vol. 18, no. 2, pp. 982–993, 2021.

J. Acharya, Z. Sun, and H. Zhang, "Hadamard response: Estimating distributions privately, efficiently, and with little communication," in *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and M. Sugiyama, Eds., vol. 89. PMLR, 2019, pp. 1120–1129.

B. Ghazi, N. Golowich, R. Kumar, R. Pagh, and A. Velingker, "On the power of multiple anonymous messages: Frequency estimation and selection in the shuffle model of differential privacy," in *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part III*, ser. Lecture Notes in Computer Science, A. Canteaut and F. Standaert, Eds., vol. 12698. Springer, 2021, pp. 463–488.

M. Charikar, K. C. Chen, and M. Farach-Colton, "Finding frequent items in data streams," in *Automata, Languages and Programming, 29th International Colloquium, ICALP 2002, Malaga, Spain, July 8-13, 2002, Proceedings*, ser. Lecture Notes in Computer Science, P. Widmayer, F. T. Ruiz, R. M. Bueno, M. Hennessy, S. J. Eidenbenz, and R. Conejo, Eds., vol. 2380. Springer, 2002, pp. 693–703.

Q. Luo, Y. Wang, and K. Yi, "Frequency estimation in the shuffle model with (almost) a single message," *CoRR*, vol. abs/2111.06833, 2021.

B. Balle, J. Bell, A. Gascón, and K. Nissim, "The privacy blanket of the shuffle model," in *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part II*, ser. Lecture Notes in Computer Science, A. Boldyreva and D. Micciancio, Eds., vol. 11693. Springer, 2019, pp. 638–667.

J. Audibert, R. Munos, and C. Szepesvári, "Exploration-exploitation tradeoff using variance estimates in multi-armed bandits," *Theor. Comput. Sci.*, vol. 410, no. 19, pp. 1876–1902, 2009.

L. Devroye and G. Lugosi, *Combinatorial methods in density estimation*, ser. Springer series in statistics. Springer, 2001.

R. Motwani and P. Raghavan, *Randomized Algorithms*. Cambridge University Press, 1995.

I. O. Tolstikhin, "Concentration inequalities for samples without replacement," *Theory of Probability & Its Applications*, vol. 61, no. 3, pp. 462–481, 2017.

Supplementary Material

The supplementary material is organized as follows:

1. In Section 6, we list the concentration inequalities applied in our proofs.
2. In Section 7, we provide the detailed proofs for Section 2.
3. In Section 8, we provide the detailed proofs for Section 3.
4. In Section 9, we provide the detailed proofs for Section 4.

6 Concentration Inequalities

Fact 6.1 (Chernoff bound (Mitzenmacher and Upfal, 2017))**.** *Let $X_1, \dots, X_n$ be independent 0-1 random variables. Let $X = \sum_{i \in [n]} X_i$ and $\mu = \mathbb{E}[X]$. Then for every $\delta > 0$,*

$$\Pr[X \geq (1 + \delta)\mu] \leq \left(\frac{e^{\delta}}{(1 + \delta)^{1 + \delta}} \right)^{\mu} .$$

Similarly, for every $\delta \in (0, 1)$

$$\Pr[X \leq (1 - \delta)\mu] \leq \left(\frac{e^{-\delta}}{(1 - \delta)^{1 - \delta}} \right)^{\mu} .$$

Fact 6.2 (Bernstein's Inequality (Audibert et al., 2009))**.** *Let $X_1, \dots, X_n$ be independent real-valued random variables such that $|X_i| \leq c$ with probability one. Let $S_n = \sum_{i \in [n]} X_i$ and $\mathbb{V}ar[S_n] = \sum_{i \in [n]} \mathbb{V}ar[X_i^2]$. Then for all $\beta \in (0, 1)$,*

$$|S_n - \mathbb{E}[S_n]| \leq \sqrt{2 \mathbb{V}ar[S_n] \ln \frac{2}{\beta}} + \frac{2c \ln \frac{2}{\beta}}{3} ,$$

with probability at least $1 - \beta$.

Fact 6.3 (Hoeffding's Inequality (Devroye and Lugosi, 2001))**.** *Let $X_1, \dots, X_n$ be independent real-valued random variables such that that $|X_i| \in [a_i, b_i], \forall i \in [n]$ with probability one. Let $S_n = \sum_{i \in [n]} X_i$, then for every $\eta \geq 0$:*

$$\Pr[S_n - \mathbb{E}[S_n] \geq \eta] \leq \exp\left(- \frac{2\eta^2}{\sum_{i \in [n]} (b_i - a_i)^2} \right) , \text{ and}$$

$$\Pr[\mathbb{E}[S_n] - S_n \geq \eta] \leq \exp\left(- \frac{2\eta^2}{\sum_{i \in [n]} (b_i - a_i)^2} \right) .$$

Definition 6.4 (Martingale (Motwani and Raghavan, 1995; Mitzenmacher and Upfal, 2017))**.** A sequence of random variables $Y_0, \dots, Y_n$ is a martingale with respect to the sequence $X_0, \dots, X_n$ if, for all $i \geq 0$, the following conditions hold: i) Y_i is a function of $X_0, \dots, X_i$; ii) $\mathbb{E}[|Y_i|] < \infty$; and iii) $\mathbb{E}[Y_{i+1} \mid X_0, \dots, X_i] = Y_i$.

Fact 6.5 (Azuma's Inequality (Mitzenmacher and Upfal, 2017))**.** *Let $Y_0, \dots, Y_n$ be a martingale such that*

$$A_i \le Y_i - Y_{i-1} \le A_i + c_i \,,$$

for some constants $\{c_i\}$ and for some random variables $\{A_i\}$ that may be functions of $Y_0, Y_1, \dots Y_{i-1}$. Then for all $t \ge 0$ and every $\eta > 0$,

$$\Pr[|Y_t - Y_0| \ge \eta] \le 2 \exp\left(-\frac{2\eta^2}{\sum_{i \in [t]} c_i^2}\right) .$$

Fact 6.6 ((Chung and Lu, 2006))**.** *Let the sequence of random variables $Y_0, \dots, Y_n$ be a martingale with respect to the sequence of random variables $X_0, \dots, X_n$ such that*

1. $\mathbb{V}ar[Y_i \mid X_0, \dots, X_{i-1}] \le \sigma_i^2, \forall i \in [n]$ *; and*
2. $|Y_i - Y_{i-1}| \le c, \forall i \in [n]$.

Then, we have

$$\Pr[Y_n - Y_0 \ge \eta] \le \exp\left(-\frac{\eta^2}{2\left(\sum_{i \in [n]} \sigma_i^2 + c\eta/3\right)}\right) .$$

Definition 6.7 (Lipschitz Condition)**.** A function $\Delta : \mathbb{R}^n \to \mathbb{R}$ satisfies the Lipschitz condition with bound $c \in \mathbb{R}$ if, for every $i \in [n]$ and for every sequence of values $x_1, \dots, x_n \in \mathbb{R}$ and $y_i \in \mathbb{R}$,

$$|\Delta(x_1, x_2, \dots, x_{i-1}, x_i, x_{i+1}, \dots, x_n) - \Delta(x_1, x_2, \dots, x_{i-1}, y_i, x_{i+1}, \dots, x_n)| \le c.$$

Fact 6.8 (McDiarmid's Inequality (Mitzenmacher and Upfal, 2017; Tolstikhin, 2017))**.** *Let $\Delta : \mathbb{R}^n \to \mathbb{R}$ be a function that satisfies the Lipschitz condition with bound $c \in \mathbb{R}$. Let $X_1, \dots, X_n$ be independent random variables such that $\Delta(X_1, \dots, X_n)$ is in the domain of Δ. Then for all $\eta \ge 0$,*

$$\Pr\left[\Delta(X_1, \dots, X_n) - \mathbb{E}\left[\Delta(X_1, \dots, X_n)\right] \ge \eta\right] \le \exp\left(-\frac{2\eta^2}{nc^2}\right) .$$

Definition 6.9 ((Tolstikhin, 2017))**.** Let $\mathbb{S}_n$ be the symmetric group of $[n]$ (i.e., the set of all possible permutations of $[n]$). A function $\Delta : \mathbb{S}_n \to \mathbb{R}$, is called (n_1, n_2)-symmetric with respect to permutations if, for each permutation $\boldsymbol{x} \in \mathbb{S}_n$, $\Delta(\boldsymbol{x})$ does not change its value under the change of order of the first n_1 and/or last $n_2 = n - n_1$ coordinates of $\boldsymbol{x}$. For brevity, we call these functions (n_1, n_2)-symmetric functions.

Fact 6.10 (McDiarmid's Inequality with respect to permutations (Tolstikhin, 2017))**.** *Let $\Delta : \mathbb{S}_n \to \mathbb{R}$ be an (n_1, n_2)-symmetric function for which there exists a constant $c > 0$ such that $|\Delta(\boldsymbol{x}) - \Delta(\boldsymbol{x}_{i,j})| \le c$ for all $\boldsymbol{x} \in \mathbb{S}_n, i \in \{1, \dots, n_1\}, j \in \{n_1 + 1, \dots, n\}$, where the permutation $\boldsymbol{x}_{i,j}$ is obtained from $\boldsymbol{x}$ by transposition of its i^{th} and j^{th} coordinates. Let $\mathbf{X}$ be a vector of random permutation chosen uniformly from a symmetric permutation group of the set $[n]$. Then for every $\eta > 0$,*

$$\Pr[\Delta(\mathbf{X}) - \mathbb{E}[\Delta(\mathbf{X})] \ge \eta] \le \exp\left(-\frac{2\eta^2}{n_1 c^2}\left(\frac{n - 1/2}{n - n_1}\right)\left(1 - \frac{1}{2\max\{n_1, n_2\}}\right)\right) .$$

7 Proofs For Section 2

This section is organized as follows:

1. In Section 7.1, we provide the detailed proof for Fact 2.1.
2. In Section 7.2, we provide the detailed proof for Corollary 2.3.

7.1 Hadamard Randomized Response (HRR)

Fact 2.1 (Algorithm **HRR** (Nguyên et al., 2016; Cormode et al., 2019))**.** *Let $\mathcal{U}$ be a set users each holding an element from some finite domain $\mathcal{D}$. There exists an ε-locally differentially private frequency oracle, **HRR**, such that the following holds. Fix some query element $v \in \mathcal{D}$ for **HRR**. With probability at least $1 - \beta'$, **HRR** returns a frequency estimate $\hat{f}_{\mathcal{U}}[v]$ satisfying*

$$\left| \hat{f}_{\mathcal{U}}[v] - f_{\mathcal{U}}[v] \right| \in O\left((1/\varepsilon) \cdot \sqrt{|\mathcal{U}| \cdot \ln(1/\beta')} \right) .$$

Each user in $\mathcal{U}$ requires $\tilde{O}(1)$ memory, takes $\tilde{O}(1)$ running time and reports only 1 bit to the server. The server processes the reports in $\tilde{O}(|\mathcal{U}| + |\mathcal{D}|)$ time and $O(|\mathcal{D}|)$ memory, and answers a query in $\tilde{O}(1)$ time. The $\tilde{O}$ notation hides logarithmic factors in $|\mathcal{U}|$, $|\mathcal{D}|$ and $1/\beta'$.

7.1.1 The Hadamard Matrix

The main vehicle for the **HRR** algorithm is the Hadamard matrix. In this section, we provide its definition, and prove some of its important properties.

Definition 7.1 (Hadamard Matrix)**.** The Hadamard matrix is defined recursively for a parameter, m, that is a power of two: $\mathrm{H}_1 = [1]$ and $\mathrm{H}_m = \begin{bmatrix} \mathrm{H}_{m/2} & \mathrm{H}_{m/2} \\ \mathrm{H}_{m/2} & -\mathrm{H}_{m/2} \end{bmatrix}$. For example, $\mathrm{H}_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$, and $\mathrm{H}_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$.

Observe that a Hadamard matrix is symmetric. We list here some other important properties of Hadamard matrix.

Fact 7.2. *The columns of the Hadamard matrix are mutually orthogonal.*

Proof of Fact 7.2 . We prove this by induction on the size of m. For H_1, this is trivially true. Suppose this holds for $\mathrm{H}_{m/2}$. For every $i \in [m/2]$, let $\boldsymbol{x}_i \in \mathbb{R}^{m/2}$ be the i^{th} column of $\mathrm{H}_{m/2}$. By the induction hypothesis, for all $i, j \in [m/2]$, if $i \neq j$, then $\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle = 0$.

Consider H_m. For each $i \in [m]$, define $\boldsymbol{y}_i \in \mathbb{R}^m$ to be the i^{th} column of H_m. Further, define

$$c(i) \doteq \begin{cases} i, & \text{if } i \leq m/2\,, \\ i - m/2 & \text{if } i > m/2\,. \end{cases} \quad \text{and} \quad s(i) \doteq \begin{cases} 1, & \text{if } i \leq (m/2)\,, \\ -1 & \text{if } i > (m/2)\,. \end{cases}$$

Note that for each $i \in [m]$, we have $c(i) \in [m/2]$. By the definition of H_m, it holds that for all $i, j \in [m]$,

$$\boldsymbol{y}_i = \begin{bmatrix} \boldsymbol{x}_{c(i)} \\ s(i) \cdot \boldsymbol{x}_{c(i)} \end{bmatrix} \quad \text{and } \boldsymbol{y}_j = \begin{bmatrix} \boldsymbol{x}_{c(j)} \\ s(j) \cdot \boldsymbol{x}_{c(j)} \end{bmatrix} .$$

Hence,

$$\langle \boldsymbol{y}_i, \boldsymbol{y}_j \rangle = \left\langle \boldsymbol{x}_{c(i)}, \boldsymbol{x}_{c(j)} \right\rangle + s(i)s(j) \left\langle \boldsymbol{x}_{c(i)}, \boldsymbol{x}_{c(j)} \right\rangle .$$

If $i \neq j$, then there are two possible cases: 1) $i \neq j + m/2$ and $j \neq i + m/2$, then by the induction hypothesis, $\left\langle \boldsymbol{x}_{c(i)}, \boldsymbol{x}_{c(j)} \right\rangle = 0$; and 2) $i = j + m/2$ or $j = i + m/2$, then $s(i)s(j) = -1$. In both cases, $\langle \boldsymbol{y}_i, \boldsymbol{y}_j \rangle = 0$ holds. □

Fact 7.3. *For $0 \leq i, j < m$, the $(i+1, j+1)$-th entry $\mathrm{H}_m[i+1, j+1]$ of the Hadamard matrix can be computed in $O(\log m)$ time (both the rows and columns of H_m are indexed from 1 to m). In particular, let the vectors $(i)_{\log m}, (j)_{\log m} \in \{0,1\}^{\log m}$ be the $\log m$-bit binary representation of i and j, respectively. Then $\mathrm{H}_m[i+1, j+1] = (-1)^{\langle (i)_{\log m}, (j)_{\log m} \rangle}$, where $\langle (i)_{\log m}, (j)_{\log m} \rangle$ is the dot product between $(i)_{\log m}$ and $(j)_{\log m}$.*

Proof of Fact 7.3 . We prove the Theorem by induction. The claim can be verified manually for H_1 and H_2. Suppose this holds for $\mathrm{H}_{m/2}$; to prove it true for H_m, recall that the recursive definition $\mathrm{H}_m = \begin{bmatrix} \mathrm{H}_{m/2} & \mathrm{H}_{m/2} \\ \mathrm{H}_{m/2} & -\mathrm{H}_{m/2} \end{bmatrix}$ divides H_m into four sub-matrices. For $0 \le i, j < m$, let the vectors $(i)_{\log m}, (j)_{\log m} \in \{0,1\}^{\log m}$ be the $\log m$-bit binary representations of i and j, respectively. Let $b_i = (i)_{\log m}[1]$ be the highest bit and $\boldsymbol{s}_i = (i)_{\log m}[2 : \log m]$ be the last $(\log m) - 1$ bits of $(i)_{\log m}$, respectively. Similarly, we can define $b_j = (j)_{\log m}[1]$ and $\boldsymbol{s}_j = (j)_{\log m}[2 : \log m]$. Now,

$$(i)_{\log m} = [b_i, \boldsymbol{s}_i], \qquad (j)_{\log m} = [b_j, \boldsymbol{s}_j].$$

Consider the $(i+1, j+1)$-th entry of H_m. Our goal is prove that

$$\mathrm{H}_m[i+1, j+1] = (-1)^{\langle (i)_{\log m}, (j)_{\log m} \rangle} = (-1)^{b_i b_j + \langle \boldsymbol{s}_i, \boldsymbol{s}_j \rangle}.$$

Observe that $b_i b_j = 1$ if the $(i+1, j+1)$-th entry belongs to the lower right sub-matrix and $b_i b_j = 0$ otherwise. By definition of H_m, the sub-matrix this entry belongs to can be written as $(-1)^{b_i b_j} \mathrm{H}_{m/2}$. If we also view $\boldsymbol{s}_i, \boldsymbol{s}_j \in \{0,1\}^{(\log m)-1}$ as integers in $[0, m/2)$, then $(\boldsymbol{s}_i + 1, \boldsymbol{s}_j + 1)$ is the pair of indexes of the entry inside the sub-matrix. By the induction hypothesis, the value of the $(\boldsymbol{s}_i + 1, \boldsymbol{s}_j + 1)$ entry of the matrix $(-1)^{b_i b_j} \mathrm{H}_{m/2}$ is given by $(-1)^{b_i b_j + \langle \boldsymbol{s}_i, \boldsymbol{s}_j \rangle}$, which finishes the proof. □

Fact 7.4 (Fast Hadamard Transform)**.** *For all $\boldsymbol{x} \in \mathbb{R}^m$, there is a standard divide-and-conquer algorithm that computes the multiplication $\mathrm{H}_m \, \boldsymbol{x}$ (*equivalently, $\mathrm{H}_m^T \boldsymbol{x}$, as H_m is symmetric*) in $O(m \log m)$ time and $O(m)$ memory.*

Proof of Fact 7.4 . Let $\boldsymbol{x}_1 \in \mathbb{R}^{m/2}$ be the first $m/2$ entries, and $\boldsymbol{x}_2 \in \mathbb{R}^{m/2}$ be the second $m/2$ entries of $\boldsymbol{x}$ respectively. Define $\boldsymbol{y}_1 = \mathrm{H}_{m/2} \, \boldsymbol{x}_1$ and $\boldsymbol{y}_2 = \mathrm{H}_{m/2} \, \boldsymbol{x}_2$. Then

$$\mathrm{H}_m \, \boldsymbol{x} = \begin{bmatrix} \mathrm{H}_{m/2} & \mathrm{H}_{m/2} \\ \mathrm{H}_{m/2} & -\mathrm{H}_{m/2} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{bmatrix} = \begin{bmatrix} \mathrm{H}_{m/2} \, \boldsymbol{x}_1 + \mathrm{H}_{m/2} \, \boldsymbol{x}_2 \\ \mathrm{H}_{m/2} \, \boldsymbol{x}_1 - \mathrm{H}_{m/2} \, \boldsymbol{x}_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{y}_1 + \boldsymbol{y}_2 \\ \boldsymbol{y}_1 - \boldsymbol{y}_2 \end{bmatrix}.$$

Let $T(m)$ be the time to compute $\mathrm{H}_m \, \boldsymbol{x}$. Computing $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$ takes time $2 \cdot T(m/2)$. Computing $\boldsymbol{y}_1 + \boldsymbol{y}_2$ and $\boldsymbol{y}_1 - \boldsymbol{y}_2$ takes time $O(m)$. Therefore, $T(m) = 2 \cdot T(m/2) + O(m)$. Solving the recursion gives $T(m) = O(m \log m)$. □

7.1.2 The Algorithm

The algorithm relies on a Hadamard matrix H_m with $m = 2^{\lceil \log |\mathcal{D}| \rceil}$ (hence $|\mathcal{D}| \le m < 2|\mathcal{D}|$), and assigns each element $v \in \mathcal{D}$ the v^{th} column of H_m. For a user $u \in \mathcal{U}$, it is said to be assigned to the v^{th} column, if its element $v^{(u)} = v$. The problem of estimating the frequency of a element $v \in \mathcal{D}$ reduces to estimating the number of users in $\mathcal{U}$ assigned to the v^{th} column. When the value of m is clear from context, we omit subscript m and write Hadamard matrix H_m as H.

7.1.3 Client Side

The client-side algorithm is described in Algorithm 2. Each user receives (from the server) a row index r of the Hadamard matrix, and privacy parameter ε. Its own element v is the column index. It returns the value of $\mathrm{H}[r, v]$, but flipped with probability $1/(e^{\varepsilon} + 1)$. Multiplying $\mathrm{H}[r, v]$ by a Rademacher random variable b that equals 1 with probability $e^{\varepsilon}/(e^{\varepsilon} + 1)$ and -1 with probability $1/(e^{\varepsilon} + 1)$ achieves the flip. It returns the one-bit result to the server.

Algorithm 2 HRR-Client $\mathcal{A}_{\text{HRR-client}}$

```
Require: Row index $r \in [m]$; privacy parameter $\varepsilon$.
Let $v \in \mathcal{D}$ be the user's element.
Sample $b \in \{-1, 1\}$, which is $+1$ with probability $e^{\varepsilon}/(e^{\varepsilon} + 1)$.
return $\omega \leftarrow b \cdot \mathrm{H}[r, v]$.
```

Fact 7.5 (Running Time and Memory Usage)**.** *Algorithm 2 has running time $\tilde{O}(1)$ and memory usage $\tilde{O}(1)$.*

Proof of Fact 7.5. According to Fact 7.3, the entry $\mathrm{H}[r, v]$ can be computed in $O(\log m) \subseteq O(\log |\mathcal{D}|) \subseteq \tilde{O}(1)$ time and $O(\log m) \subseteq O(\log |\mathcal{D}|) \subseteq \tilde{O}(1)$ memory. □

Fact 7.6 (Privacy Guarantee)**.** *Algorithm 2 is ε-locally differentially private.*

Proof of Fact 7.6. We need to prove that the output distribution of $\mathcal{A}_{\text{HRR-client}}$ deviates little with the value $v \in \mathcal{D}$, the user's element. To explicitly state the dependence of $\mathcal{A}_{\text{HRR-client}}$ on v, we write its output as $\mathcal{A}_{\text{HRR-client}}(r, \varepsilon; v)$. It suffices to prove that $\forall v, v' \in \mathcal{D}$, the output distributions of $\mathcal{A}_{\text{HRR-client}}(r, \varepsilon; v)$ and $\mathcal{A}_{\text{HRR-client}}(r, \varepsilon; v')$ are similar. There are only two possible outputs, namely, $\{-1, 1\}$. Let b and b' be the Rademacher random variables generated by $\mathcal{A}_{\text{HRR-client}}(r, \varepsilon; v)$ and $\mathcal{A}_{\text{HRR-client}}(r, \varepsilon; v')$ respectively. Then

$$\begin{aligned}\Pr[\mathcal{A}_{\text{HRR-client}}(r, \varepsilon; v) = 1] &= \Pr[b \cdot \mathrm{H}[r, v] = 1] \le e^{\varepsilon}/(e^{\varepsilon}+1)\,,\\ \Pr[\mathcal{A}_{\text{HRR-client}}(r, \varepsilon; v') = 1] &= \Pr[b' \cdot \mathrm{H}[r, v'] = 1] \ge 1/(e^{\varepsilon}+1)\,.\end{aligned}$$

Hence, $\Pr[\mathcal{A}_{\text{HRR-client}}(r, \varepsilon; v) = 1] \le e^{\varepsilon} \cdot \Pr[\mathcal{A}_{\text{HRR-client}}(r, \varepsilon; v') = 1]$. By Definition 1.1, the algorithm $\mathcal{A}_{\text{HRR-client}}$ is ε-differentially private. □

7.1.4 Server Side

The server-side algorithm is described in Algorithm 3. Its input comprises the set of users, $\mathcal{U}$, their elements' domain, $\mathcal{D}$, and privacy parameter ε. The server maintains a vector $\boldsymbol{\omega} \in \mathbb{R}^m$, which is initialized with all zeros. For each user $u \in \mathcal{U}$, the server samples an integer $r^{(u)} \in [m]$ independently and uniformly at random. Then it invokes $\mathcal{A}_{\text{HRR-client}}(r^{(u)}, \varepsilon)$ by sending $r^{(u)}$ and ε to user u. On receiving user u's response, $\omega^{(u)}$, the server increases the $(r^{(u)})^{\text{th}}$ entry of $\boldsymbol{\omega}$ by $(e^{\varepsilon}+1)/(e^{\varepsilon}-1) \cdot \omega^{(u)}$. Finally, it returns a vector $\hat{f}_{\mathcal{U}} = \mathrm{H}^T \boldsymbol{\omega} \in \mathbb{R}^m$. Note that the dimension of $\hat{f}_{\mathcal{U}}$ is m, which could be larger than $|\mathcal{D}|$: we use only the first $|\mathcal{D}|$ entries of $\hat{f}_{\mathcal{U}}$.

Algorithm 3 HRR-Server $\mathcal{A}_{\text{HRR-server}}$

```
Require: A set of users $\mathcal{U}$; element domain $\mathcal{D}$; privacy parameter $\varepsilon$.
1: Set $m \leftarrow 2^{\lceil \log |\mathcal{D}| \rceil}$, $\boldsymbol{\omega} \leftarrow \{0\}^m$.
2: for $u \in \mathcal{U}$ do
3:     $r^{(u)} \leftarrow$ uniform random integer from $[m]$.
4:     $\omega^{(u)} \leftarrow \mathcal{A}_{\text{HRR-client}}(r^{(u)}, \varepsilon)$.
5:     $\boldsymbol{\omega}[r^{(u)}] \leftarrow \boldsymbol{\omega}[r^{(u)}] + (e^{\varepsilon}+1)/(e^{\varepsilon}-1) \cdot \omega^{(u)}$.
6: return $\hat{f}_{\mathcal{U}} \leftarrow \mathrm{H}^T \boldsymbol{\omega}$.
```

Fact 7.7 (Running Time and Memory Usage)**.** *Algorithm 3 has running time $\tilde{O}(|\mathcal{D}| + |\mathcal{U}|)$ and memory usage $O(|\mathcal{D}|)$.*

Proof of Fact 7.7. The server needs memory of size $O(m) \subseteq O(|\mathcal{D}|)$ to store the vector $\boldsymbol{\omega}$. Processing responses from the users in $\mathcal{U}$ takes time $O(|\mathcal{U}|)$. By Fact 7.4, $\mathrm{H}^T\boldsymbol{\omega}$ can be computed in $O(m \log m) \subseteq \tilde{O}(|\mathcal{D}|)$ time, with memory usage $O(m) \subseteq O(|\mathcal{D}|)$. Hence the overall running time is $\tilde{O}(|\mathcal{D}| + |\mathcal{U}|)$ and memory usage is $O(|\mathcal{D}|)$. □

Remark 7.1. Via the fast Hadamard transform (Fact 7.4), the server computes $\hat{f}_{\mathcal{U}} \leftarrow \mathrm{H}^T \boldsymbol{\omega}$ (Algorithm 3, line 6) in $\tilde{O}(|\mathcal{D}|)$ time. Then for each $v \in \mathcal{D}$, if its frequency is queried, the server can return $\hat{f}_{\mathcal{U}}[v]$ in $O(1)$ time. There is another version of **HRR** that omits line 6. It has server running time $\tilde{O}(|\mathcal{U}|)$. However, when it answers a frequency query for some $v \in \mathcal{D}$, it needs to compute $\hat{f}_{\mathcal{U}}[v] = (\mathrm{H}^T \boldsymbol{\omega})[v]$ on the fly, which requires $\tilde{O}(\min\{m, |\mathcal{U}|\})$ time.

7.1.5 Utility Guarantee

We have proven that the client-side algorithm is ε-locally differentially private, and analyzed the running time and memory usage of both client-side and server-side algorithms. In this section, we discuss their utility guarantees.

Fact 7.8 (Expectation)**.** *Let $\hat{f}_{\mathcal{U}}$ be the estimate vector returned by Algorithm 3. Then for all $v \in \mathcal{D}$, $\hat{f}_{\mathcal{U}}[v]$ is an unbiased estimator of $f_{\mathcal{U}}[v]$.*

Proof of Fact 7.8. For each $i \in [m]$, let $\boldsymbol{e}_i$ be the i^{th} standard basis vector. For each user $u \in \mathcal{U}$, let $\boldsymbol{c}^{(u)} \doteq \mathrm{H}\, \boldsymbol{e}_{v^{(u)}}$ be the column assigned to u, i.e., the $(v^{(u)})^{\text{th}}$ column of the Hadamard matrix H, and let $b^{(u)}$ be the Rademacher random variable generated when algorithm $\mathcal{A}_{\text{HRR-client}}$ is invoked for user u. Let $C_{\varepsilon} \doteq (e^{\varepsilon}+1)/(e^{\varepsilon}-1)$. By algorithm $\mathcal{A}_{\text{HRR-client}}$, the response from user u can be expressed as $\omega^{(u)} = b^{(u)} \cdot \boldsymbol{c}^{(u)}[r^{(u)}]$. When the server receives the response $\omega^{(u)}$, the update of $\boldsymbol{\omega}$ can be rewritten as

$$\boldsymbol{\omega} \leftarrow \boldsymbol{\omega} + C_\varepsilon \cdot \omega^{(u)} \cdot \boldsymbol{e}_{r^{(u)}} \,.$$

Hence $\boldsymbol{\omega} = \sum_{u \in \mathcal{U}} C_\varepsilon \cdot \omega^{(u)} \cdot \boldsymbol{e}_{r^{(u)}} = \sum_{u \in \mathcal{U}} C_\varepsilon \cdot b^{(u)} \cdot \boldsymbol{c}^{(u)}[r^{(u)}] \cdot \boldsymbol{e}_{r^{(u)}}$, where $\boldsymbol{e}_{r^{(u)}}$ is the $(r^{(u)})^{\text{th}}$ standard basis vector. Let $\boldsymbol{c}_v \doteq \mathrm{H}\ \boldsymbol{e}_v$ be the v^{th} column of H. Since $\hat{f}_{\mathcal{U}} = \mathrm{H}^T\ \boldsymbol{\omega}$, we have

$$\hat{f}_{\mathcal{U}}[v] = \langle \boldsymbol{c}_v, \boldsymbol{\omega} \rangle = \sum_{u \in \mathcal{U}} C_\varepsilon \cdot b^{(u)} \cdot \boldsymbol{c}^{(u)}[r^{(u)}] \cdot \langle \boldsymbol{c}_v, \boldsymbol{e}_{r^{(u)}} \rangle = \sum_{u \in \mathcal{U}} C_\varepsilon \cdot b^{(u)} \cdot \boldsymbol{c}^{(u)}[r^{(u)}] \cdot \boldsymbol{c}_v[r^{(u)}] \,.$$

By the independence of $b^{(u)}$ and $r^{(u)}$, and by linearity of expectation, we have

$$\mathbb{E}\left[\hat{f}_{\mathcal{U}}[v]\right] = \sum_{u \in \mathcal{U}} C_\varepsilon \cdot \mathbb{E}[b^{(u)}] \cdot \mathbb{E}\left[\boldsymbol{c}^{(u)}[r^{(u)}] \cdot \boldsymbol{c}_v[r^{(u)}]\right] = \sum_{u \in \mathcal{U}} \mathbb{E}\left[\boldsymbol{c}^{(u)}[r^{(u)}] \cdot \boldsymbol{c}_v[r^{(u)}]\right].$$

The second equality follows from $\mathbb{E}[b^{(u)}] = 1 \cdot e^\varepsilon/(e^\varepsilon + 1) + (-1) \cdot 1/(e^\varepsilon + 1) = 1/C_\varepsilon$.

As $r^{(u)}$ is sampled uniformly from $[m]$, it holds that

$$\sum_{u \in \mathcal{U}} \mathbb{E}\left[\boldsymbol{c}^{(u)}[r^{(u)}] \cdot \boldsymbol{c}_v[r^{(u)}]\right] = \sum_{u \in \mathcal{U}} \frac{1}{m} \cdot \sum_{j=1}^{m} \left(\boldsymbol{c}^{(u)}[j] \cdot \boldsymbol{c}_v[j]\right) = \sum_{u \in \mathcal{U}} \frac{1}{m} \cdot \left\langle \boldsymbol{c}^{(u)}, \boldsymbol{c}_v \right\rangle = \sum_{u \in \mathcal{U}} \mathbb{1}[v = v^{(u)}] \,.$$

The final equality follows from the orthogonality of columns of H, and that $\langle \boldsymbol{c}_v, \boldsymbol{c}_v \rangle = m$. We conclude that

$$\mathbb{E}\left[\hat{f}_{\mathcal{U}}[v]\right] = \sum_{u \in \mathcal{U}} \mathbb{1}[v = v^{(u)}] = f_{\mathcal{U}}[v].$$

□

Fact 7.9 (Confidence Interval)**.** *For a fixed $v \in \mathcal{D}$ and for all $\beta' \in (0,1)$, with probability at least $1 - \beta'$, it holds that*

$$|\hat{f}_{\mathcal{U}}[v] - f_{\mathcal{U}}[v]| \in O\left((1/\varepsilon) \cdot \sqrt{|\mathcal{U}| \cdot \ln(1/\beta')}\right) .$$

***Proof of Fact 7.9*.** For each $u \in \mathcal{U}$, define $Z^{(u)} \doteq C_\varepsilon \cdot b^{(u)} \cdot \boldsymbol{c}^{(u)}[r^{(u)}] \cdot \boldsymbol{c}_v[r^{(u)}]$. The $\{Z^{(u)}\}$ are independent random variables in the range of $[-C_\varepsilon, C_\varepsilon]$. As $\hat{f}_{\mathcal{U}}[v] = \sum_{u \in \mathcal{U}} Z^{(u)}$ and $\mathbb{E}\left[\hat{f}_{\mathcal{U}}[v]\right] = f_{\mathcal{U}}[v]$, by Hoeffding's inequality (Fact 6.3), for all $\eta > 0$,

$$\Pr\left[\left|\hat{f}_{\mathcal{U}}[v] - f_{\mathcal{U}}[v]\right| \geq \eta\right] \leq 2 \exp\left(-\frac{2\eta^2}{\sum_{u \in \mathcal{U}} (C_\varepsilon - (-C_\varepsilon))^2}\right) .$$

If we upper bound the failure probability with β', we obtain that $\eta \leq C_\varepsilon \cdot \sqrt{2|\mathcal{U}| \ln(2/\beta')}$. Noting that $C_\varepsilon \in O(1/\varepsilon)$ for $\varepsilon \in O(1)$ finishes the proof. □

7.2 Proof of Corollary 2.3

Corollary 2.3. *Let $\varepsilon \in O(1)$. Every ε-**LDP** frequency oracle algorithm achieving estimation error λ with probability at least $1 - \beta'$ must have*

$$\lambda \in \Omega\left((1/\varepsilon) \cdot \sqrt{|\mathcal{U}| \cdot \ln(1/\beta')}\right).$$

Proof of Corollary 2.3. Let $\beta' = \beta/d$ be the failure probability of the frequency oracle algorithm, and suppose by contradiction that $\lambda \in o((1/\varepsilon) \cdot \sqrt{|\mathcal{U}| \ln(1/\beta')})$ for this algorithm. If so, we could query it for the frequency of all elements in the domain $\mathcal{D}$. By a union bound, we could thus construct a succinct histogram with failure probability at most β and with error for every element in $o((1/\varepsilon) \cdot \sqrt{|\mathcal{U}| \ln(1/\beta')})$. Since the latter bound is in, $o((1/\varepsilon) \cdot \sqrt{|\mathcal{U}| \ln(|\mathcal{D}|/\beta)})$, this contradicts Fact 2.2. □

8 Proofs For Section 3

This section is organized as follows:

1. In Section 8.1, we provide the detailed proof for Theorem 3.1.
2. In Section 8.2, we provide the detailed proof for Theorem 3.3.
3. In Section 8.3, we provide the proof for Lemma 8.1, which we rely on to prove Theorem 3.3.
4. In Section 8.4, we provide the proof for Lemma 8.2, which we rely on to prove Theorem 3.3.

8.1 Theorem 3.1

The properties of Theorem 3.1 have been established implicitly in Section 3. Here we show how to put the pieces together explicitly.

Theorem 3.1 (Sketching Framework). *For every $\beta' \in (0,1)$, $\mathcal{A}_{oracle}$ can be converted into an ε-**LDP** frequency oracle, with server running time $\tilde{O}(\Phi_{time}(|\mathcal{U}|, \sqrt{|\mathcal{U}|}))$ and memory usage $\tilde{O}(\Phi_{mem}(|\mathcal{U}|, \sqrt{|\mathcal{U}|}))$. Fix an element $v \in \mathcal{D}$ to be given as a query to the new algorithm. With probability at least $1-\beta'$, it returns an estimate $\hat{f}_{\mathcal{U}}[v]$ satisfying*

$$\left|\hat{f}_{\mathcal{U}}[v] - f_{\mathcal{U}}[v]\right| \in O\left((1/\varepsilon) \cdot \sqrt{|\mathcal{U}| \cdot \ln(1/\beta')}\right).$$

Proof of Theorem 3.1. Since Algorithm 1 (Sketching Framework) invokes hash functions to reduce the domain size from $|\mathcal{D}|$ to $m \in O(\sqrt{|\mathcal{U}|})$, it follows that it has server running time $\tilde{O}(\Phi_{time}(|\mathcal{U}|, \sqrt{|\mathcal{U}|}))$ and memory usage $\tilde{O}(\Phi_{mem}(|\mathcal{U}|, \sqrt{|\mathcal{U}|}))$. The privacy guarantee follows from that Algorithm 1 partitions the set of users $\mathcal{U}$ into subsets, and invokes $\mathcal{A}_{oracle}$ for each subset. Therefore, each user participates in only one copy of $\mathcal{A}_{oracle}$. As $\mathcal{A}_{oracle}$ is ε differentially private, so is Algorithm 1. Finally, the utility guarantee follows from Corollary 3.4. □

This finishes the proof of Theorem 3.1. In the next section, we discuss Theorem 3.3.

8.2 Theorem 3.3

Theorem 3.3 *With probability at least $1-\beta'/4$, it holds that $|\mathbb{G}d\mathbb{S}et_0| > (1-1/8)k$. And for each $v \in \mathcal{D}$ and each $j \in [3]$, with probability at least $1-\beta'/4$, it holds that $|\mathbb{G}d\mathbb{S}et_j(v)| > (1-1/8)k$.*

Proof of Theorem 3.3. We need to prove the Theorem for $\mathbb{G}d\mathbb{S}et_0, \mathbb{G}d\mathbb{S}et_1(v), \mathbb{G}d\mathbb{S}et_2(v), \mathbb{G}d\mathbb{S}et_3(v)$, separately. As they are easier, we first bound the sizes of $\mathbb{G}d\mathbb{S}et_2(v)$ and $\mathbb{G}d\mathbb{S}et_3(v)$.

Bounding the Size of $\mathbb{G}d\mathbb{S}et_2(v)$.

Fix some $v \in \mathcal{D}$. Recall that $\lambda_2(j,v) \doteq |kf_{\mathcal{U}_i,h_i}[h(v)] - kf_{\mathcal{U}i}[v]|$. Via the definition of $\mathbb{G}d\mathbb{S}et_2(v)$, it can be rewritten as

$$\mathbb{G}d\mathbb{S}et_2(v) = \left\{ i \in [k] : |kf_{\mathcal{U}_i,h_i}[h(v)] - kf_{\mathcal{U}i}[v]| \in O\left(\frac{1}{\varepsilon} \cdot \sqrt{k \cdot |\mathcal{U}_i| \ln \frac{1}{\beta'}}\right) \right\}.$$

Observe that for each $i \in [k]$, the (scaled) errors $|\lambda_2(i,v)|/k = |f_{\mathcal{U}_i,h_i}[h(v)] - f_{\mathcal{U}i}[v]|$ result from hash collisions. Consider an fixed $i \in [k]$. For each user $u \in \mathcal{U}_i$, define the indicator random variable $X_u = \mathbb{1}[h_i(v^{(u)}) = h_i(v)]$ for the event $h_i(v^{(u)}) = h_i(v)$. If $v^{(u)} = v$, it always holds that $X_u = 1$. Otherwise, as h_i is a pairwise-independent hash function, $\Pr[X_u = 1] = 1/m$.

By definition, $f_{\mathcal{U}_i,h_i}[h_i(v)] = \sum_{u \in \mathcal{U}_i} X_u$. Therefore,

$$\mathbb{E}\left[|f_{\mathcal{U}_i,h_i}[h_i(v)] - f_{\mathcal{U}i}[v]|\right] = \mathbb{E}\left[\left|\sum_{u \in \mathcal{U}_i} X_u - f_{\mathcal{U}i}[v]\right|\right] = \mathbb{E}\left[\sum_{u \in \mathcal{U}_i, v^{(u)} \neq v} X_u\right].$$

By linearity of expectation, we have

$$\mathbb{E}\left[|f_{\mathcal{U}_i,h_i}[h_i(v)] - f_{\mathcal{U}_i}[v]| \right] = \frac{|u \in \mathcal{U}_i, v^{(u)} \neq v|}{m} \leq \frac{|\mathcal{U}_i|}{m}.$$

By Markov's inequality,

$$\Pr\left[|f_{\mathcal{U}_i,h_i}[h_i(v)] - f_{\mathcal{U}_i}[v]| \geq \frac{1}{\varepsilon}\sqrt{\frac{1}{k}\cdot|\mathcal{U}_i| \ln\frac{4}{\beta'}} \right] \leq \frac{|\mathcal{U}_i|/m}{(1/\varepsilon)\sqrt{(1/k)\cdot|\mathcal{U}_i|\ln(4/\beta')}} = \frac{\varepsilon\sqrt{k|\mathcal{U}_i|}}{m\sqrt{\ln(4/\beta')}}.$$

Recall that Algorithm 1 initializes $k = C_K \cdot \ln(4/\beta')$ and $m = 8e^2 \cdot \sqrt{C_K} \cdot \varepsilon \cdot \sqrt{|\mathcal{U}|}$ for some constant C_K. The upper bound on the probability simplifies to $\sqrt{|\mathcal{U}_i|} \,/\, (8e^2 \sqrt{|\mathcal{U}|})$. Using that $|\mathcal{U}| \geq |\mathcal{U}_i|$, this upper bound further simplifies to $1/(8e^2)$.

For each $i \in [k]$, define the indicator random variable

$$Y_i \doteq \mathbb{1}\left[|kf_{\mathcal{U}_i,h_i}[h(v)] - kf_{\mathcal{U}_i}[v]| \geq \frac{1}{\varepsilon}\sqrt{k\cdot|\mathcal{U}_i|\ln\frac{4}{\beta'}} \right]$$

for the event $|kf_{\mathcal{U}_i,h_i}[h(v)] - kf_{\mathcal{U}_i}[v]| \geq (1/\varepsilon)\sqrt{k\cdot|\mathcal{U}_i|\ln(4/\beta')}$. Let $Y \doteq \sum_{i\in[k]} Y_i$. We have $\mathbb{E}[Y_i] \leq 1/(8e^2)$, and $\mu \doteq \mathbb{E}[Y] \leq k/(8e^2)$. As the $h_1, .., h_k$ are chosen independently, the $\{Y_i\}$ are independent. Via Chernoff bound (Fact 6.1),

$$\Pr\left[Y \geq \frac{k}{8}\right] = \Pr\left[Y \geq \left(1 + \left(\frac{k}{8\mu} - 1\right)\right)\mu\right] \leq \left(\frac{\exp\left(k/(8\mu) - 1\right)}{(k/(8\mu))^{k/(8\mu)}}\right)^{\mu} = \exp\left(\frac{k}{8} - \mu - \frac{k}{8}\ln\frac{k}{8\mu}\right).$$

For $\mu \leq k/(8e^2)$, the function $-\mu - (k/8)\ln(k/(8\mu)) = -\mu - (k/8)\ln(k/8) + (k/8)\ln\mu$ is maximized when $\mu = k/(8e^2)$. Therefore,

$$\Pr\left[Y \geq \frac{k}{8}\right] \leq \exp\left(\frac{k}{8} - \frac{k}{8e^2} - \frac{k}{8}\ln e^2\right) = \exp\left(-\frac{k}{8}\left(1 + \frac{1}{e^2}\right)\right).$$

Recall that $k = C_K \cdot \ln(4/\beta')$. If we set $C_K = 8$, then we get $\Pr[Y \geq k/8] \leq \beta'/4$.

■

Bounding the Size of $\mathbb{G}d\mathbb{S}et_3(v)$.

Via the assumption of $\mathcal{A}_{oracle}$, for $i \in [k]$, with probability at most $1/(8e^2)$,

$$|\hat{f}_{\mathcal{U}_i,h_i}[h_i(v)] - f_{\mathcal{U}_i,h_i}[h_i(v)]| \notin O\left(\frac{1}{\varepsilon}\sqrt{|\mathcal{U}_i|\cdot\ln(8e^2)}\right).$$

Scaling both sides by a factor of k, we get

$$|\lambda_3(i,v)| = |k\cdot\hat{f}_{\mathcal{U}_i,h_i}[h_i(v)] - k\cdot f_{\mathcal{U}_i,h_i}[h_i(v)]| \notin O\left(\frac{1}{\varepsilon}\sqrt{k^2|\mathcal{U}_i|\cdot\ln(8e^2)}\right).$$

Replacing one factor k with $C_K \cdot \ln(4/\beta')$, we have $|\lambda_3(i,v)| \notin O((1/\varepsilon)\sqrt{k|\mathcal{U}_j|\cdot\ln(1/\beta')})$. Since for each $i \in [k]$, the event happens independently, the probability that there are more than $k/8$ choices of $i \in [k]$ for which this event happens is at most

$$\binom{k}{k/8}\left(\frac{1}{8e^2}\right)^{k/8} \leq \left(\frac{ek}{k/8}\right)^{k/8}\left(\frac{1}{8e^2}\right)^{k/8} = \left(\frac{1}{e}\right)^{k/8},$$

where the first inequality follows from that $\binom{k}{k/8} \leq \frac{k^{k/8}}{(k/8)!}$ and that $\frac{(k/8)^{k/8}}{(k/8)!} \leq e^{k/8}$. Recall that $k = C_K \cdot \ln(4/\beta')$. If we set $C_K = 8$, then we get $(1/e)^{k/8} \leq \beta'/4$.

■

To bound the sizes of $\mathbb{G}d\mathbb{S}et_0$ and $\mathbb{G}d\mathbb{S}et_1(v)$, we need the following lemmas.

Lemma 8.1. *Let* $\Delta_0 \doteq \sum_{i \in [k]} \left| |\mathcal{U}_i| - |\mathcal{U}|/k \right|$. $\exists\, C_0 > 0$*, s.t., with probability* $1 - \beta'/4$*:* $\Delta_0 \le C_0 \sqrt{|\mathcal{U}| \ln(4/\beta')}$.

Lemma 8.2. *Let* $\Delta_1 \doteq \sum_{i \in [k]} \|f_{\mathcal{U}i} - f_{\mathcal{U}}/k\|_2$*, where* $\|f_{\mathcal{U}i} - f_{\mathcal{U}}/k\|_2 \doteq \sqrt{\sum_{v' \in \mathcal{D}} (f_{\mathcal{U}i}[v'] - f_{\mathcal{U}}[v']/k)^2}$*. There exists some constant* $C_1 > 0$*, s.t., with probability* $1 - \beta'/4$*:* $\Delta_1 \le C_1 \sqrt{|\mathcal{U}| \ln(4/\beta')}$.

We need to prove the lemmas for both *independent partitioning* and *permutation partitioning*. The proofs are technical, so we defer them to the end of the proof. For now, we show how to put them together to bound the sizes of $\mathbb{G}d\mathbb{S}et_0$ and $\mathbb{G}d\mathbb{S}et_1(v)$.

Bounding the Size of $\mathbb{G}d\mathbb{S}et_0$.

By Lemma 8.1, with probability at least $1 - \beta'/4$, it holds that $\Delta_0 \le C_0 \sqrt{|\mathcal{U}| \ln(4/\beta')}$ for some constant C_0. Therefore,

$$k \cdot \Delta_0 = \sum_{i \in [k]} \left| k \cdot |\mathcal{U}_i| - |\mathcal{U}| \right| \le k C_0 \sqrt{|\mathcal{U}| \ln \frac{4}{\beta'}} \,.$$

By a counting argument, the number of $i \in [k]$, such that $\left| k \cdot |\mathcal{U}_i| - |\mathcal{U}| \right| \ge 8 C_0 \sqrt{|\mathcal{U}| \ln(4/\beta')}$ is bounded by $(1/8)k$. This implies that for at least $(1 - 1/8)k$ of the $i \in [k]$, we have

$$\left| k \cdot |\mathcal{U}_i| - |\mathcal{U}| \right| \le 8 C_0 \sqrt{|\mathcal{U}| \ln \frac{4}{\beta'}} \,.$$

By the assumption that $|\mathcal{U}| \ge \ln(4/\beta')$, we get $\left| k \cdot |\mathcal{U}_i| - |\mathcal{U}| \right| \in \Theta(|\mathcal{U}|)$.

■

Bounding the Size of $\mathbb{G}d\mathbb{S}et_1(v)$.

By Lemma 8.2, with probability at least $1 - \beta'/4$, it holds that $\Delta_1 \le C_1 \sqrt{|\mathcal{U}| \ln(4/\beta')}$ for some constant C_1. Hence,

$$k \cdot \Delta_1 = k \sum_{i \in [k]} \|f_{\mathcal{U}i} - f_{\mathcal{U}}/k\|_2 \le k C_1 \sqrt{|\mathcal{U}| \ln \frac{4}{\beta'}}.$$

By a counting argument, the number of $i \in [k]$, such that $k \left\| f_{\mathcal{U}i} - f_{\mathcal{U}}/k \right\|_2 \ge 8 C_1 \sqrt{|\mathcal{U}| \ln(4/\beta')}$ is bounded by $(1/8)k$. This implies that for at least $(1 - 1/8)k$ of the $i \in [k]$, we have

$$\left| \lambda_1(i, v) \right| = \left| k f_{\mathcal{U}i}[v] - f_{\mathcal{U}}[v] \right| \le k \left\| f_{\mathcal{U}i} - f_{\mathcal{U}}/k \right\|_2 \le 8 C_1 \sqrt{|\mathcal{U}| \ln(4/\beta')} \,.$$

which finishes the proof.

■

□

In the following two sections, we prove Lemma 8.1 and Lemma 8.2 respectively.

8.3 Bounding Δ_0

Lemma 8.1. *Let* $\Delta_0 \doteq \sum_{i \in [k]} \left| |\mathcal{U}_i| - |\mathcal{U}|/k \right|$. $\exists\, C_0 > 0$*, s.t., with probability* $1 - \beta'/4$*:* $\Delta_0 \le C_0 \sqrt{|\mathcal{U}| \ln(4/\beta')}$.

The lemma holds trivially for **permutation partitioning**, as in such case it holds that $|\mathcal{U}_i| = |\mathcal{U}|/k$ and $\Delta_0 = 0$. We need to prove the lemma for **independent partitioning**.

8.3.1 Proof of Lemma 8.1 for Independent Partitioning

Without loss of generality, assume that $\mathcal{U} = \{1, 2, .., |\mathcal{U}|\}$. For each $u \in \mathcal{U}$, let $X_u \in [k]$ be the index of the subset that user u belongs to. Let $\mathbf{X} \doteq (X_1, \ldots, X_{|\mathcal{U}|})$: by definition, for each $i \in [k]$, $\Pr[X_u = i] = 1/k$. The set $\mathcal{U}_i$ can be represented as

$$\mathcal{U}_i \doteq \{u \in \mathcal{U} : X_u = i\} \,.$$

Now, Δ_0 can be rewritten as

$$\Delta_0 = \sum_{i \in [k]} \sqrt{\left(\sum_{u \in \mathcal{U}} \mathbb{1}\left[X_u = i\right] - |\mathcal{U}|/k \right)^2},$$

where $\mathbb{1}\left[X_u = i\right]$ is the indicator random variable for the event $X_u = i$. Hence, Δ_0 is a random variables that depends on $\mathbf{X}$. We write Δ_0 explicitly as $\Delta_0(X_1, \ldots, X_{|\mathcal{U}|})$ or $\Delta_0(\mathbf{X})$ when necessary. For a sequence of values $\boldsymbol{x} = \{x_1, \ldots, x_{|\mathcal{U}|}\} \in [k]^{|\mathcal{U}|}$, we use $\Delta_0(x_1, \ldots, x_{|\mathcal{U}|})$ or $\Delta_0(\boldsymbol{x})$ to denote the value of Δ_0, when $\mathbf{X} = \boldsymbol{x}$. Observe that

$$\Delta_0 = \Delta_0 - \mathbb{E}[\Delta_0] + \mathbb{E}[\Delta_0].$$

In order to upper bound Δ_0, we can upper bound both $\Delta_0 - \mathbb{E}[\Delta_0]$ and $\mathbb{E}[\Delta_0]$ superlatively. In particular, we will prove that 1) with probability at least $1 - \beta'/4$, it holds that $\Delta_0 - \mathbb{E}[\Delta_0] \le \sqrt{2|\mathcal{U}| \ln(4/\beta')}$; 2) $\mathbb{E}[\Delta_0] \le \sqrt{k|\mathcal{U}|}$. Substituting $k = C_K \cdot \ln(4/\beta')$, we get that, with probability at least $1 - \beta'/4$,

$$\Delta_0 \le \sqrt{2|\mathcal{U}| \ln(4/\beta')} + \sqrt{k|\mathcal{U}|} = \sqrt{2|\mathcal{U}| \ln(4/\beta')} + \sqrt{C_K |\mathcal{U}| \ln(4/\beta')}.$$

As we set $C_K = 8$ in the proof of Theorem 3.3 in Section 8.2, the RHS simplifies to $\Delta_0 \le 3\sqrt{2|\mathcal{U}| \ln(4/\beta')}$.

Step 1: Bounding $\Delta_0 - \mathbb{E}[\Delta_0]$.

We upper bound it by McDiarmid's Inequality (Fact 6.8). We will prove that Δ_0 satisfies Lipschitz condition (Definition 6.7) with bound 2, i.e., for all $u \in \mathcal{U}$, and every sequence of values $\boldsymbol{x} = \{x_1, \ldots, x_u, \ldots, x_{|\mathcal{U}|}\} \in [k]^{|\mathcal{U}|}$ and $x'_u \in [k]$,

$$|\Delta_0(x_1, \ldots, x_u, \ldots, x_{|\mathcal{U}|}) - \Delta_0(x_1, \ldots, x'_u, \ldots, x_{|\mathcal{U}|})| \le 2\,. \tag{3}$$

Then by McDiarmid's Inequality (Fact 6.8),

$$\Pr\left[\Delta_0 - \mathbb{E}[\Delta_0] \ge \sqrt{2|\mathcal{U}| \ln \frac{4}{\beta'}}\right] \le \exp\left(- \frac{2\left(\sqrt{2|\mathcal{U}| \ln(4/\beta')}\right)^2}{|\mathcal{U}| \cdot 4} \right) \le \beta'/4\,.$$

Proof of Inequality (3). Define a random vector in $\mathbb{R}^k$ that depends on $\mathbf{X}$ as

$$\boldsymbol{\omega}(\mathbf{X}) \doteq \left(\sum_{u \in \mathcal{U}} \mathbb{1}\left[X_u = 1\right] - \frac{|\mathcal{U}|}{k}, \ \ldots, \ \sum_{u \in \mathcal{U}} \mathbb{1}\left[X_u = k\right] - \frac{|\mathcal{U}|}{k} \right).$$

For a sequence of values $\boldsymbol{x} = \{x_1, \ldots, x_u, \ldots, x_{|\mathcal{U}|}\} \in [k]^{|\mathcal{U}|}$, let $\boldsymbol{\omega}(\boldsymbol{x})$ be the vector of $\boldsymbol{\omega}(\mathbf{X})$ when $\mathbf{X} = \boldsymbol{x}$. By its definition, $\Delta_0(\boldsymbol{x})$ equals $\|\boldsymbol{\omega}(\boldsymbol{x})\|_1$, the ℓ_1 norm of $\boldsymbol{\omega}(\boldsymbol{x})$.

Consider a fixed $u \in \mathcal{U}$. Let $\boldsymbol{x}' = \{x_1, \ldots, x'_u, \ldots, x_{|\mathcal{U}|}\}$ be the sequence obtained by replacing x_u with x'_u. The inequality (3) clearly holds when $x_u = x'_u$. Now, suppose that $x_u \neq x_u$. Then $\boldsymbol{\omega}(\boldsymbol{x})$ and $\boldsymbol{\omega}(\boldsymbol{x}')$ differ in only two coordinates, each by 1. Specifically, $\boldsymbol{\omega}(\boldsymbol{x}) - \boldsymbol{\omega}(\boldsymbol{x}') = \boldsymbol{e}_{x_u} - \boldsymbol{e}_{x'_u}$, where $\boldsymbol{e}_{x_u}$ and $\boldsymbol{e}_{x'_u}$ are the $(x_u)^{(th)}$ and the $(x'_u)^{(th)}$ standard basis vectors in $\mathbb{R}^k$, respectively. By the triangle inequality,

$$\left| \|\boldsymbol{\omega}(\boldsymbol{x})\|_1 - \|\boldsymbol{\omega}(\boldsymbol{x}')\|_1 \right| \le \|\boldsymbol{e}_{x_u} - \boldsymbol{e}_{x'_u}\|_1 = 2.$$

Step 2: Bounding $\mathbb{E}[\Delta_0]$.

By Jensen's inequality, it holds that

$$\mathbb{E}\left[\Delta_0\right] = \sum_{i \in [k]} \mathbb{E}\left[\sqrt{\left(\sum_{u \in \mathcal{U}} \mathbb{1}\left[X_u = i\right] - |\mathcal{U}|/k \right)^2} \right] \le \sum_{i \in [k]} \sqrt{\mathbb{E}\left[\left(\sum_{u \in \mathcal{U}} \mathbb{1}\left[X_u = i\right] - |\mathcal{U}|/k \right)^2 \right]}.$$

Fix an $i \in [k]$. For each $u \in \mathcal{U}$, define the indicator random variable $Z^{(u)} \doteq \mathbb{1}[X_u = i]$ for the event $u \in \mathcal{U}_i$. Then $\Pr[Z^{(u)} = 1] = 1/k$ and $\Pr[Z^{(u)} = 0] = 1 - 1/k$. Further $|\mathcal{U}_i| = \sum_{u \in \mathcal{U}} Z^{(u)}$, is a sum of $|\mathcal{U}|$ independent random variables and has expectation $|\mathcal{U}|/k$. Hence,

$$\mathbb{E}\left[\left(\sum_{u\in\mathcal{U}} Z^{(u)} - |\mathcal{U}|/k\right)^2\right] = \mathbb{V}\text{ar}\left[|\mathcal{U}_i|\right] = \sum_{u\in\mathcal{U}} \mathbb{V}\text{ar}\left[Z^{(u)}\right] \le \frac{|\mathcal{U}|}{k}\,.$$

Therefore,

$$\mathbb{E}\left[\Delta_0\right] \le \sum_{i\in[k]} \sqrt{\mathbb{E}\left[\left(\sum_{u\in\mathcal{U}} \mathbb{1}\left[X_u = i\right] - |\mathcal{U}|/k\right)^2\right]} \le \sum_{i\in[k]} \sqrt{\frac{|\mathcal{U}|}{k}} = \sqrt{k|\mathcal{U}|}\,.$$

□

This finishes the proof of Lemma 8.1. Next, we prove Lemma 8.2.

8.4 Bounding Δ_1

Lemma 8.2. *Let* $\Delta_1 \doteq \sum_{i\in[k]} \|f_{\mathcal{U}^i} - f_{\mathcal{U}}/k\|_2$*, where* $\|f_{\mathcal{U}^i} - f_{\mathcal{U}}/k\|_2 \doteq \sqrt{\sum_{v'\in\mathcal{D}}(f_{\mathcal{U}^i}[v'] - f_{\mathcal{U}}[v']/k)^2}$*. There exists some constant* $C_1 > 0$*, s.t., with probability* $1 - \beta'/4$*:* $\Delta_1 \le C_1\sqrt{|\mathcal{U}|\ln(4/\beta')}$*.*

We need to prove the lemma for both **independent partitioning** and **permutation partitioning**.

8.4.1 Proof of Lemma 8.2 for Independent Partitioning

Without loss of generality, assume that $\mathcal{U} = \{1, 2, .., |\mathcal{U}|\}$. For each $u \in \mathcal{U}$, let $X_u \in [k]$ be the index of the subset that user u belongs to. Let $\mathbf{X} \doteq (X_1, \ldots, X_{|\mathcal{U}|})$: by definition, for each $i \in [k]$, $\Pr[X_u = i] = 1/k$. The set $\mathcal{U}_i$ can be represented as

$$\mathcal{U}_i \doteq \{u \in \mathcal{U} : X_u = i\}\,.$$

For each $v \in \mathcal{D}$, we have

$$f_{\mathcal{U}^i}[v] = \sum_{u\in\mathcal{U}_i} \mathbb{1}\left[v^{(u)} = v\right],$$

where $\mathbb{1}\left[v^{(u)} = v\right]$ is the indicator random variable for the event $v^{(u)} = v$. Therefore,

$$\left\|f_{\mathcal{U}^i} - \frac{f_{\mathcal{U}}}{k}\right\|_2 = \sqrt{\sum_{v\in\mathcal{D}}\left(f_{\mathcal{U}^i}[v] - \frac{f_{\mathcal{U}}[v]}{k}\right)^2}$$

is a random variables that depends on $\mathbf{X}$. We write $\|f_{\mathcal{U}^i} - f_{\mathcal{U}}/k\|_2$ explicitly as $\|f_{\mathcal{U}^i} - f_{\mathcal{U}}/k\|_2\,(X_1, \ldots, X_{|\mathcal{U}|})$ or $\|f_{\mathcal{U}^i} - f_{\mathcal{U}}/k\|_2\,(\mathbf{X})$ when necessary. For a sequence of values $\boldsymbol{x} = \{x_1, \ldots, x_{|\mathcal{U}|}\} \in [k]^{|\mathcal{U}|}$, we use $\|f_{\mathcal{U}^i} - f_{\mathcal{U}}/k\|_2\,(x_1, \ldots, x_{|\mathcal{U}|})$ or $\|f_{\mathcal{U}^i} - f_{\mathcal{U}}/k\|_2\,(\boldsymbol{x})$ to denote the value of $\|f_{\mathcal{U}^i} - f_{\mathcal{U}}/k\|_2\,(\mathbf{X})$, when $\mathbf{X} = \boldsymbol{x}$.

Moreover, as $\Delta_1 = \sum_{i\in[k]} \|f_{\mathcal{U}^i} - f_{\mathcal{U}}/k\|_2$, it is also a random variables that depends on $\mathbf{X}$. We write Δ_1 explicitly as $\Delta_1(X_1, \ldots, X_{|\mathcal{U}|})$ or $\Delta_1(\mathbf{X})$ when necessary. For a sequence of values $\boldsymbol{x} = \{x_1, \ldots, x_{|\mathcal{U}|}\} \in [k]^{|\mathcal{U}|}$, we use $\Delta_1(x_1, \ldots, x_{|\mathcal{U}|})$ or $\Delta_1(\boldsymbol{x})$ to denote the value of Δ_1, when $\mathbf{X} = \boldsymbol{x}$.

Observe that

$$\Delta_1 = \Delta_1 - \mathbb{E}[\Delta_1] + \mathbb{E}[\Delta_1].$$

In order to upper bound Δ_1, we can upper bound both $\Delta_1 - \mathbb{E}[\Delta_1]$ and $\mathbb{E}[\Delta_1]$ superlatively. In particular, we will prove that 1) with probability at least $1 - \beta'/4$, it holds that $\Delta_1 - \mathbb{E}[\Delta_1] \le \sqrt{2|\mathcal{U}|\ln(4/\beta')}$; 2) $\mathbb{E}[\Delta_1] \le \sqrt{k|\mathcal{U}|}$. Substituting $k = C_K \cdot \ln(4/\beta')$, we get that, with probability at least $1 - \beta'/4$,

$$\Delta_1 \le \sqrt{2|\mathcal{U}|\ln(4/\beta')} + \sqrt{k|\mathcal{U}|} = \sqrt{2|\mathcal{U}|\ln(4/\beta')} + \sqrt{C_K|\mathcal{U}|\ln(4/\beta')}.$$

As we set $C_K = 8$ in the proof of Theorem 3.3 in Section 8.2, the RHS simplifies to $\Delta_1 \le 3\sqrt{2|\mathcal{U}|\ln(4/\beta')}$.

Step 1: Bounding $\Delta_1 - \mathbb{E}[\Delta_1]$.

We upper bound it by McDiarmid's Inequality (Fact 6.8). We will prove that Δ_1 satisfies Lipschitz condition (Definition 6.7) with bound 2, i.e., for all $u \in \mathcal{U}$, and every sequence of values $\boldsymbol{x} = \{x_1, \dots, x_u, \dots, x_{|\mathcal{U}|}\} \in [k]^{|\mathcal{U}|}$ and $x'_u \in [k]$,

$$|\Delta_1(x_1, \dots, x_u, \dots, x_{|\mathcal{U}|}) - \Delta_1(x_1, \dots, x'_u, \dots, x_{|\mathcal{U}|})| \le 2\,. \tag{4}$$

Then by McDiarmid's Inequality (Fact 6.8),

$$\Pr\left[\Delta_1 - \mathbb{E}[\Delta_1] \ge \sqrt{2|\mathcal{U}| \ln \frac{4}{\beta'}}\right] \le \exp\left(-\frac{2\left(\sqrt{2|\mathcal{U}| \ln(4/\beta')}\right)^2}{|\mathcal{U}| \cdot 4}\right) \le \beta'/4\,.$$

Proof of Inequality (4).

Consider a fixed $u \in \mathcal{U}$. Let $\boldsymbol{x}' = \{x_1, \dots, x'_u, \dots, x_{|\mathcal{U}|}\}$ be the sequence obtained by replacing x_u with x'_u. The inequality (4) clearly holds when $x_u = x'_u$. It is left to consider the case when $x_u \ne x'_u$. To simplify the notation, denote $j = x_u$ and $\ell = x'_u$. Via the definition that $\Delta_1 = \sum_{i \in [k]} \left\| f_{\mathcal{U}^i} - \frac{f_{\mathcal{U}}}{k} \right\|_2$, the $\Delta_1(\boldsymbol{x})$ and $\Delta_1(\boldsymbol{x}')$ differ only in two terms. Specifically,

$$\Delta_1(\boldsymbol{x}) - \Delta_1(\boldsymbol{x}') = \left\| f_{\mathcal{U}^j} - \frac{f_{\mathcal{U}}}{k} \right\|_2 (\boldsymbol{x}) - \left\| f_{\mathcal{U}^j} - \frac{f_{\mathcal{U}}}{k} \right\|_2 (\boldsymbol{x}') + \left\| f_{\mathcal{U}^\ell} - \frac{f_{\mathcal{U}}}{k} \right\|_2 (\boldsymbol{x}) - \left\| f_{\mathcal{U}^\ell} - \frac{f_{\mathcal{U}}}{k} \right\|_2 (\boldsymbol{x}')$$

For each $v \in \mathcal{D}$, define $f_{\mathcal{U}_j, \boldsymbol{x}}[v] \doteq |\{u \in \mathcal{U}_j : v^{(u)} = v\}|$ to be the frequency of v in the set $\{v^{(u)} : u \in \mathcal{U}_j\}$, when $\mathbf{X} = \boldsymbol{x}$. Let $f_{\mathcal{U}_j, \boldsymbol{x}} \doteq \left(f_{\mathcal{U}_j, \boldsymbol{x}}[v] : v \in \mathcal{D}\right)$ be the frequency vector when $\mathbf{X} = \boldsymbol{x}$. Similarly, define $f_{\mathcal{U}_j, \boldsymbol{x}'}$ to be the frequency vector when $\mathbf{X} = \boldsymbol{x}'$. Further, let $f_{\mathcal{U}_\ell, \boldsymbol{x}}$ and $f_{\mathcal{U}_\ell, \boldsymbol{x}'}$ be the frequency vectors defined on $\mathcal{U}_\ell$, when $\mathbf{X} = \boldsymbol{x}$ and $\mathbf{X} = \boldsymbol{x}'$ respectively. Recall that $f_{\mathcal{U}} = \left(f_{\mathcal{U}}[v] : v \in \mathcal{D}\right)$ denotes the frequency vector defined on the entire user set $\mathcal{U}$.

Let $v^{(u)}$ be the data of user u. When the value of $\mathbf{X}$ changes from $\boldsymbol{x}$ to $\boldsymbol{x}'$, the subset that user u belongs to switches from $\mathcal{U}_j$ to $\mathcal{U}_\ell$. The frequency of $v^{(u)}$ in $\mathcal{U}_j$ decreases by 1, and such frequency in $\mathcal{U}_\ell$ increases by 1. Therefore,

$$f_{\mathcal{U}_j, \boldsymbol{x}} - f_{\mathcal{U}_j, \boldsymbol{x}'} = \boldsymbol{e}_{v^{(u)}}\,, \quad f_{\mathcal{U}_\ell, \boldsymbol{x}} - f_{\mathcal{U}_\ell, \boldsymbol{x}'} = -\boldsymbol{e}_{v^{(u)}}\,.$$

where $\boldsymbol{e}_{v^{(u)}}$ is the $v^{(u)}$-th standard basis vector in $\mathbb{R}^{|\mathcal{D}|}$. As the norm $\|\cdot\|_2$ satisfies the triangle inequality, we obtain

$$\left\| f_{\mathcal{U}_j, \boldsymbol{x}} - \frac{f_{\mathcal{U}}}{k} \right\|_2 - \left\| f_{\mathcal{U}_j, \boldsymbol{x}'} - \frac{f_{\mathcal{U}}}{k} \right\|_2 \le \|\boldsymbol{e}_{v^{(u)}}\|_2 = 1\,, \quad \left\| f_{\mathcal{U}_\ell, \boldsymbol{x}} - \frac{f_{\mathcal{U}}}{k} \right\|_2 - \left\| f_{\mathcal{U}_\ell, \boldsymbol{x}'} - \frac{f_{\mathcal{U}}}{k} \right\|_2 \le \|-\boldsymbol{e}_{v^{(u)}}\|_2 = 1\,.$$

Therefore, $|\Delta_1(\boldsymbol{x}) - \Delta_1(\boldsymbol{x}')| \le 2$.

Step 2: Bounding $\mathbb{E}[\Delta_1]$. By linearity of expectation,

$$\mathbb{E}[\Delta_1] = \sum_{i \in [k]} \mathbb{E}\left[\left\| f_{\mathcal{U}^i} - \frac{f_{\mathcal{U}}}{k} \right\|_2\right].$$

For a fixed $i \in [k]$, by Jensen's inequality, it holds that

$$\mathbb{E}\left[\left\| f_{\mathcal{U}^i} - \frac{f_{\mathcal{U}}}{k} \right\|_2\right] = \mathbb{E}\left[\sqrt{\sum_{v \in \mathcal{D}} \left(f_{\mathcal{U}^i}[v] - \frac{f_{\mathcal{U}}[v]}{k}\right)^2}\right] \le \sqrt{\sum_{v \in \mathcal{D}} \mathbb{E}\left[\left(f_{\mathcal{U}^i}[v] - \frac{f_{\mathcal{U}}[v]}{k}\right)^2\right]}\,.$$

For a fixed $v \in \mathcal{D}$, define $\mathcal{U}[v] \doteq \{u \in \mathcal{U} : v^{(u)} = v\}$ as the subset of users in $\mathcal{U}$ holding element v. It holds that $|\mathcal{U}[v]| = f_{\mathcal{U}}[v]$. For each $u \in \mathcal{U}[v]$, define the indicator random variable $Z^{(u)} \doteq \mathbb{1}[X_u = i]$ for the event $u \in \mathcal{U}_i$. Then $\Pr[Z^{(u)} = 1] = 1/k$ and $\Pr[Z^{(u)} = 0] = 1 - 1/k$. Then $f_{\mathcal{U}^i}[v] = \sum_{u \in \mathcal{U}[v]} Z^{(u)}$, is a sum of $f_{\mathcal{U}}[v]$ independent random variables with expectation $f_{\mathcal{U}}[v]/k$. Hence,

$$\mathbb{E}\left[\left(f_{\mathcal{U}^i}[v] - \frac{f_{\mathcal{U}}[v]}{k}\right)^2\right] = \mathbb{V}\mathrm{ar}\left[f_{\mathcal{U}^i}[v]\right] = \sum_{u \in \mathcal{U}[v]} \mathbb{V}\mathrm{ar}\left[Z^{(u)}\right] \le \frac{f_{\mathcal{U}}[v]}{k}\,.$$

Therefore,

$$\mathbb{E}\left[\left\|f_{\mathcal{U}^i} - \frac{f_{\mathcal{U}}}{k}\right\|_2\right] \leq \sqrt{\sum_{v \in \mathcal{D}} \mathbb{E}\left[\left(f_{\mathcal{U}^i}[v] - \frac{f_{\mathcal{U}}[v]}{k}\right)^2\right]} \leq \sqrt{\sum_{v \in \mathcal{D}} \frac{f_{\mathcal{U}}[v]}{k}} = \sqrt{\frac{|\mathcal{U}|}{k}}\,.$$

Finally, summing over all $i \in [k]$, we obtain

$$\mathbb{E}[\Delta_1] = \sum_{i \in [k]} \mathbb{E}\left[\left\|f_{\mathcal{U}^i} - \frac{f_{\mathcal{U}}}{k}\right\|_2\right] \leq \sqrt{|\mathcal{U}|k}\,.$$

□

8.4.2 Proof of Lemma 8.2 for Permutation Partitioning

We need to bound $\Delta_1 = \big(\Delta_1 - \mathbb{E}[\Delta_1]\big) + \mathbb{E}[\Delta_1]$; we bound $\Delta_1 - \mathbb{E}[\Delta_1]$ by $2\sqrt{|\mathcal{U}| \ln \frac{2}{\beta'}}$ and $\mathbb{E}[\Delta_1]$ by $\sqrt{k|\mathcal{U}|}$.

Without loss of generality, assume that $\mathcal{U} = \{1, 2, .., |\mathcal{U}|\}$. Let $\mathbf{X} = (X_1, X_2, \ldots, X_{|\mathcal{U}|})$ be a random permutation of $\mathcal{U}$, i.e., one chosen uniformly at random from the set of all possible permutation of $\mathcal{U}$. Note that $X_1, \ldots, X_{|\mathcal{U}|}$ are dependent random variables.

For each $j \in [k]$, by the way we generate $\mathcal{U}_j$, it has size $|\mathcal{U}|/k$ and $\mathcal{U}_j = \{X_i : (j-1) \cdot |\mathcal{U}|/k + 1 \leq i \leq j \cdot |\mathcal{U}|/k\}$. For every $v \in \mathcal{D}$, we can write

$$f_{\mathcal{U}^j}[v] = \sum_{X_i \in \mathcal{U}_j} \mathbb{1}\left[v^{(X_i)} = v\right]\,,$$

where $\mathbb{1}\left[v^{(X_i)} = v\right]$ is the indicator random variable for the event $v^{(X_i)} = v$. Therefore,

$$\left\|f_{\mathcal{U}^j} - \frac{f_{\mathcal{U}}}{k}\right\|_2 = \sqrt{\sum_{v \in \mathcal{D}} \left(f_{\mathcal{U}^j}[v] - \frac{f_{\mathcal{U}}[v]}{k}\right)^2} = \sqrt{\sum_{v \in \mathcal{D}} \left(\sum_{X_i \in \mathcal{U}_j} \mathbb{1}\left[v^{(X_i)} = v\right] - \frac{f_{\mathcal{U}}[v]}{k}\right)^2}\,.$$

Now, $\Delta_1 = \sum_{j \in [k]} \left\|f_{\mathcal{U}^j} - \frac{f_{\mathcal{U}}}{k}\right\|_2$ is a function that depends on $\mathbf{X}$; we write Δ_1 explicitly as $\Delta_1(\mathbf{X})$ or $\Delta_1(X_1, \ldots, X_{|\mathcal{U}|})$ when necessary. For a sequence of values $\boldsymbol{x} = \{x_1, \ldots, x_{|\mathcal{U}|}\}$, we use $\Delta_1(x_1, \ldots, x_{|\mathcal{U}|})$ or $\Delta_1(\boldsymbol{x})$ to denote the value of Δ_1, when $\mathbf{X} = \boldsymbol{x}$.

Martingale Construction.

We will apply a martingale concentration inequality (Fact 6.5) for the proof. First, to construct a martingale that satisfies Definition 6.4, we introduce a dummy variable $X_0 \equiv 0$. For each $0 \leq i \leq |\mathcal{U}|$, let $\mathbf{S}_i$ be shorthand for $(X_0, \ldots, X_i)$, and define

$$Y_i \doteq \mathbb{E}[\Delta_1 \mid \mathbf{S}_i]\,.$$

Clearly Y_i is a function of $X_0, \ldots, X_i$ and $\mathbb{E}[|Y_i|] \leq \infty$. Moreover,

$$\mathbb{E}[Y_{i+1} \mid \mathbf{S}_i] = \mathbb{E}\left[\mathbb{E}\left[\Delta_1 \,|\, \mathbf{S}_i, X_{i+1}\right] | \,\mathbf{S}_i\right] = \mathbb{E}\left[\Delta_1 \,|\, \mathbf{S}_i\right] = Y_i.$$

The sequence $Y_0, \ldots, Y_n$ satisfies all conditions specified in Definition 6.4 and is a martingale. By definition, $Y_{|\mathcal{U}|} = \mathbb{E}[\Delta_1 \mid \mathbf{S}_{|\mathcal{U}|}] = \Delta_1$, as once the values of $X_1, \ldots, X_{|\mathcal{U}|}$ are determined, so is Δ_1. And we have $Y_0 = \mathbb{E}\left[\Delta_1 \,|\, X_0\right] = \mathbb{E}\left[\Delta_1\right]$, as $X_0 \equiv 0$.

Observe that

$$\Delta_1 = \Delta_1 - \mathbb{E}[\Delta_1] + \mathbb{E}[\Delta_1] = Y_{|\mathcal{U}|} - Y_0 + Y_0\,,$$

In order to upper bound Δ_1, we can upper bound both $Y_{|\mathcal{U}|} - Y_0$ and Y_0.

Step 1: Bounding $\Delta_1 - \mathbb{E}[\Delta_1]$.

We upper bound it via Azuma's Inequality (Fact 6.5). We prove that,

$$A_i \le Y_i - Y_{i-1} \le A_i + 2\sqrt{2}\,, \tag{5}$$

for some random variables $\{A_i\}$ that are functions of $X_0, \ldots, X_{i-1}$. By Azuma's inequality,

$$\Pr\left[|Y_{|\mathcal{U}|} - Y_0| \ge \left(2\sqrt{|\mathcal{U}|\ln\frac{2}{\beta'}}\right)\right] \le 2\exp\left(-\frac{2\left(2\sqrt{|\mathcal{U}|\ln(2/\beta')}\right)^2}{\sum_{i\in\mathcal{U}}(2\sqrt{2})^2}\right) \le \beta'\,.$$

Proof of Inequality (5). We prove that the gap between the upper and lower bounds on $Y_i - Y_{i-1}$ is at most $2\sqrt{2}$. By the definitions of Y_i and Y_{i-1},

$$Y_i - Y_{i-1} = \mathbb{E}[\Delta_1 \mid \mathbf{S}_i] - \mathbb{E}[\Delta_1 \mid \mathbf{S}_{i-1}]\,.$$

Let $\mathcal{U} \setminus \mathbf{S}_{i-1}$ be the set of integers in $\mathcal{U}$ that are distinct from $X_1, \ldots, X_{i-1}$. Define

$$A_i = \inf_{x\in\mathcal{U}\setminus\mathbf{S}_{i-1}} \mathbb{E}[\Delta_1 \mid \mathbf{S}_{i-1}, X_i = x] - \mathbb{E}[\Delta_1 \mid \mathbf{S}_{i-1}]\,, \text{ and } \quad B_i = \sup_{x'\in\mathcal{U}\setminus\mathbf{S}_{i-1}} \mathbb{E}[\Delta_1 \mid \mathbf{S}_{i-1}, X_i = x'] - \mathbb{E}[\Delta_1 \mid \mathbf{S}_{i-1}]\,.$$

Clearly $A_i \le Y_i - Y_{i-1} \le B_i$. We prove that $B_i - A_i \le 2\sqrt{2}$. Let $x_0 \equiv 0$, and for each $j \ge 0$, $\boldsymbol{x}_j = (x_0, x_1, \ldots, x_j)$ be the sequence that consists of a starting 0, and the first j entries of a possible permutation of $\mathcal{U}$. For each $i \ge 1$, let $\mathcal{U} \setminus \boldsymbol{x}_{i-1}$ be the set of integers in $\mathcal{U}$ that are distinct from $x_1, \ldots, x_{i-1}$. Conditioned on $\mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}$,

$$\begin{aligned} B_i - A_i &= \sup_{x'\in\mathcal{U}\setminus\boldsymbol{x}_{i-1}} \mathbb{E}[\Delta_1 \mid \mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}, X_i = x'] - \inf_{x\in\mathcal{U}\setminus\boldsymbol{x}_{i-1}} \mathbb{E}[\Delta_1 \mid \mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}, X_i = x] \\ &= \sup_{x',x\in\mathcal{U}\setminus\boldsymbol{x}_{i-1}} \left(\mathbb{E}[\Delta_1 \mid \mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}, X_i = x'] - \mathbb{E}[\Delta_1 \mid \mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}, X_i = x]\right)\,. \end{aligned}$$

It suffices to bound this for every possible sequence of $\boldsymbol{x}_{i-1}$.

Consider a fixed $i \in \mathcal{U}$ and $\boldsymbol{x}_{i-1}$. Define

$$\gamma_{x',x} \doteq \mathbb{E}[\Delta_1 \mid \mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}, X_i = x'] - \mathbb{E}[\Delta_1 \mid \mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}, X_i = x]\,.$$

Our goal is to prove for all $x', x \in \mathcal{U} \setminus \boldsymbol{x}_{i-1}$, $\gamma_{x',x'} \le 2\sqrt{2}$. It follows that $\sup_{x',x\in\mathcal{U}\setminus\boldsymbol{x}_{i-1}} \gamma_{x',x'} \le 2\sqrt{2}$. If $x' = x$, then $\gamma_{x',x'} = 0$. Suppose $x' \ne x$. As $\mathbf{X}$ is a random permutation of $\mathcal{U}$, conditioned on $\mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}$ and $X_i = x'$, with equal probability, one of the elements $X_{i+1}, \ldots, X_{|\mathcal{U}|}$ equals x. Hence,

$$\mathbb{E}[\Delta_1 \mid \mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}, X_i = x'] = \frac{1}{|\mathcal{U}| - i} \sum_{\ell=i+1}^{|\mathcal{U}|} \mathbb{E}[\Delta_1 \mid \mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}, X_i = x', X_\ell = x]\,.$$

Similarly, it holds that

$$\mathbb{E}[\Delta_1 \mid \mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}, X_i = x] = \frac{1}{|\mathcal{U}| - i} \sum_{\ell=i+1}^{|\mathcal{U}|} \mathbb{E}[\Delta_1 \mid \mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}, X_i = x, X_\ell = x']\,.$$

By triangle inequality,

$$\gamma_{x',x} = \frac{1}{|\mathcal{U}| - i} \sum_{\ell=i+1}^{|\mathcal{U}|} \left[\mathbb{E}[\Delta_1 \mid \mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}, X_i = x', X_\ell = x] - \mathbb{E}[\Delta_1 \mid \mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}, X_i = x, X_\ell = x']\right]$$

For all permutation sequences $\boldsymbol{x} = (x_1, \ldots, x_{|\mathcal{U}|})$ and for all $i \ne \ell \in \mathcal{U}$, define $\boldsymbol{x}_{i,\ell}$ to be the sequence with the values of x_i and x_ℓ being swapped. We claim it holds that

$$|\Delta_1(\boldsymbol{x}) - \Delta_1(\boldsymbol{x}_{i,\ell})| \le 2\sqrt{2}, \tag{6}$$

This proves that

$$\mathbb{E}[\Delta_1 \mid \mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}, X_i = x', X_\ell = x] - \mathbb{E}[\Delta_1 \mid \mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}, X_i = x, X_\ell = x'] \le 2\sqrt{2}, \quad \text{and} \quad \gamma_{x',x} \le 2\sqrt{2}.$$

To prove Inequality (6), recall that $\Delta_1 = \sum_{j\in[k]} \left\| f_{\mathcal{U}j} - \frac{f_{\mathcal{U}}}{k} \right\|_2$.

If $v^{(x_i)} = v^{(x_\ell)}$ or there exists some $j \in [k]$, s.t. both $x_i, x_\ell \in \mathcal{U}_j$, then the swap does not change Δ_1, and $|\Delta_1(\boldsymbol{x}) - \Delta_1(\boldsymbol{x}_{i,\ell})| = 0$.

Otherwise, $v^{(x_i)} \ne v^{(x_\ell)}$ and $x_i \in \mathcal{U}_j$, $x_\ell \in \mathcal{U}_{j'}$ for different $j, j' \in [k]$. The swap affects only $\left\| f_{\mathcal{U}j} - \frac{f_{\mathcal{U}}}{k} \right\|_2$ and $\left\| f_{\mathcal{U}j'} - \frac{f_{\mathcal{U}}}{k} \right\|_2$. Let $f_{\mathcal{U}_j,\boldsymbol{x}}$ and $f_{\mathcal{U}_j,\boldsymbol{x}_{i,\ell}}$ be the frequency vectors when $\mathbf{X} = \boldsymbol{x}$ and $\mathbf{X} = \boldsymbol{x}_{i,\ell}$, respectively. They differ in both the $(v^{(x_i)})^{\text{th}}$ and $(v^{(x_\ell)})^{\text{th}}$ coordinates, each by 1.

If we view $f_{\mathcal{U}_j,\boldsymbol{x}} - \frac{f_{\mathcal{U}}}{k}$ and $f_{\mathcal{U}_j,\boldsymbol{x}_{i,\ell}} - \frac{f_{\mathcal{U}}}{k}$ as $|\mathcal{D}|$-dimensional vectors, it holds that

$$\left(f_{\mathcal{U}_j,\boldsymbol{x}} - \frac{f_{\mathcal{U}}}{k} \right) - \left(f_{\mathcal{U}_j,\boldsymbol{x}_{i,\ell}} - \frac{f_{\mathcal{U}}}{k} \right) = -\boldsymbol{e}_{v^{(x_i)}} + \boldsymbol{e}_{v^{(x_\ell)}},$$

where $\boldsymbol{e}_{v^{(x_i)}}$ and $\boldsymbol{e}_{v^{(x_\ell)}}$ are the $v^{(x_i)}$-th and the $v^{(x_\ell)}$-th standard basis vectors in $\mathbb{R}^{|\mathcal{D}|}$ respectively. By the triangle inequality,

$$\left\| f_{\mathcal{U}_j,\boldsymbol{x}} - \frac{f_{\mathcal{U}}}{k} \right\|_2 - \left\| f_{\mathcal{U}_j,\boldsymbol{x}_{i,\ell}} - \frac{f_{\mathcal{U}}}{k} \right\|_2 \le \| -\boldsymbol{e}_{v^{(x_i)}} + \boldsymbol{e}_{v^{(x_\ell)}} \|_2 = \sqrt{2}.$$

Similarly, we can prove that the change of $\left\| f_{\mathcal{U}j'} - \frac{f_{\mathcal{U}}}{k} \right\|_2$ is bounded by $\sqrt{2}$. Therefore, $|\Delta_1(\boldsymbol{x}) - \Delta_1(\boldsymbol{x}_{i,\ell})| \le 2\sqrt{2}$.

Step 2: Bounding $\mathbb{E}[\Delta_1]$.

By linearity of expectation,

$$Y_0 = \mathbb{E}[\Delta_1] = \sum_{j\in[k]} \mathbb{E}\left[\left\| f_{\mathcal{U}j} - \frac{f_{\mathcal{U}}}{k} \right\|_2 \right].$$

For a fixed $j \in [k]$, by Jensen's inequality, it holds that

$$\mathbb{E}\left[\left\| f_{\mathcal{U}j} - \frac{f_{\mathcal{U}}}{k} \right\|_2 \right] = \mathbb{E}\left[\sqrt{ \sum_{v\in\mathcal{D}} \left(f_{\mathcal{U}j}[v] - \frac{f_{\mathcal{U}}[v]}{k} \right)^2 } \right] \le \sqrt{ \sum_{v\in\mathcal{D}} \mathbb{E}\left[\left(f_{\mathcal{U}j}[v] - \frac{f_{\mathcal{U}}[v]}{k} \right)^2 \right] }.$$

Consider a fixed $v \in \mathcal{D}$, define $\mathcal{U}[v] \doteq \{u \in \mathcal{U} : v^{(u)} = v\}$ as the set of users holding element v. It holds that $|\mathcal{U}[v]| = f_{\mathcal{U}}[v]$. For each $u \in \mathcal{U}[v]$, define the indicator random variable $Z^{(u)}$ for the event $u \in \mathcal{U}_j$. Then $\Pr[Z^{(u)} = 1] = 1/k$ and $\Pr[Z^{(u)} = 0] = 1 - 1/k$. Then

$$f_{\mathcal{U}_j}[v] = \sum_{u\in\mathcal{U}[v]} Z^{(u)},$$

is a sum of $f_{\mathcal{U}}[v]$ dependent random variables with expectation $f_{\mathcal{U}}[v]/k$. For a pair of users $u, u' \in \mathcal{U}[v], u \ne u'$, due the permutation, if u belongs to $\mathcal{U}_j$, it is less likely that u' belongs to $\mathcal{U}_j$. In particular,

$$\begin{aligned}
\mathbb{C}\mathrm{ov}\left[Z^{(u)}, Z^{(u')} \right] &= \mathbb{E}\left[Z^{(u)} \cdot Z^{(u')} \right] - \mathbb{E}\left[Z^{(u)} \right] \mathbb{E}\left[Z^{(u')} \right] \\
&= \binom{|\mathcal{U}| - 2}{|\mathcal{U}|/k - 2} \Big/ \binom{|\mathcal{U}|}{|\mathcal{U}|/k} - \left(\frac{1}{k} \right)^2 \\
&= \frac{|\mathcal{U}|/k \cdot (|\mathcal{U}|/k - 1)}{|\mathcal{U}|(|\mathcal{U}| - 1)} - \left(\frac{1}{k} \right)^2 \\
&\le 0\,.
\end{aligned}$$

Hence,

$$\begin{aligned}\mathbb{E}\left[\left(f_{\mathcal{U}^j}[v]-\frac{f_{\mathcal{U}}[v]}{k}\right)^2\right] &= \mathbb{V}\mathrm{ar}\left[f_{\mathcal{U}^j}[v]\right] = \sum_{u\in\mathcal{U}[v]}\mathbb{V}\mathrm{ar}\left[Z^{(u)}\right] + \sum_{u\neq u'\in\mathcal{U}[v]}\mathbb{C}\mathrm{ov}\left[Z^{(u)},Z^{(u')}\right] \\ &\le \sum_{u\in\mathcal{U}[v]}\mathbb{V}\mathrm{ar}\left[Z^{(u)}\right] = f_{\mathcal{U}}[v]\cdot\frac{1}{k}\cdot\left(1-\frac{1}{k}\right)\le\frac{f_{\mathcal{U}}[v]}{k}\,.\end{aligned}$$

Therefore,

$$\mathbb{E}\left[\left\|f_{\mathcal{U}^j}-\frac{f_{\mathcal{U}}}{k}\right\|_2\right]\le\sqrt{\sum_{v\in\mathcal{D}}\mathbb{E}\left[\left(f_{\mathcal{U}^j}[v]-\frac{f_{\mathcal{U}}[v]}{k}\right)^2\right]}\le\sqrt{\sum_{v\in\mathcal{D}}\frac{f_{\mathcal{U}}[v]}{k}}=\sqrt{\frac{|\mathcal{U}|}{k}}\,,$$

and

$$Y_0=\mathbb{E}[\Delta_1]=\sum_{j\in[k]}\mathbb{E}\left[\left\|f_{\mathcal{U}^j}-\frac{f_{\mathcal{U}}}{k}\right\|_2\right]\le\sqrt{|\mathcal{U}|k}\,.$$

9 Proofs For Section 4

This section is organized as follows:

1. In Section 9.1, we provide the detailed proof for Theorem 4.2.
2. In Section 9.2, we provide the detailed proof for Theorem 4.3.

9.1 Theorem 4.2

Theorem 4.2. For each $\tau \in [L]$, fix some query string $\boldsymbol{s} \in \Lambda^\tau$ for the frequency estimate. It holds that, with probability $1 - \beta'$,

$$|\hat{f}_{\mathcal{U}}[\boldsymbol{s}] - f_{\mathcal{U}}[\boldsymbol{s}]| \in O((1/\varepsilon)\sqrt{n \cdot (\log d) \cdot (\ln(1/\beta'))/\ln n})\,.$$

Proof of Theorem 4.2. Recall that for each $\tau \in [L]$ and each $\boldsymbol{s} \in \Lambda^\tau$, $f_{\mathbb{U}_\tau}[\boldsymbol{s}] \doteq |\{u \in \mathbb{U}_\tau : v^{(u)}[1:\tau] = \boldsymbol{s}\}|$ is the frequency of $\boldsymbol{s}$ in $\mathbb{U}_\tau$, and $\hat{f}_{\mathbb{U}_\tau}[\boldsymbol{s}]$ is its estimate by **HadaOracle**.

For each $\tau \in [L]$, define frequency vector $f_{\mathbb{U}_\tau} \doteq \left(f_{\mathbb{U}_\tau}[\boldsymbol{s}] : \boldsymbol{s} \in \Lambda^\tau\right)$. Denote the ℓ_2 distance between the frequency vector $f_{\mathbb{U}_\tau}$ and its expectation $f_{\mathcal{U}}/L$ as

$$\left\| f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L} \right\|_2 \doteq \sqrt{\sum_{\boldsymbol{s} \in \Lambda^\tau} \left(f_{\mathbb{U}_\tau}[\boldsymbol{s}] - \frac{f_{\mathcal{U}}[\boldsymbol{s}]}{L} \right)^2}\,.$$

To prove Theorem 4.2, we need the following lemmas.

Lemma 9.1. *For each $\tau \in [L]$, with probability $1 - \beta'$, it holds that $|\mathbb{U}_\tau| \in O(n/L)$.*

Lemma 9.2. *For each $\tau \in [L]$, with probability $1 - \beta'$, it holds that*

$$\left\| f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L} \right\|_2 \in O\left(\sqrt{\frac{n}{L} \ln \frac{1}{\beta'}} \right)\,.$$

We need to prove the lemmas for both *independent partitioning* and *permutation partitioning*. The proofs are technical, so we defer them to the end of the proof. For now, we show how to put them together to complete the proof of Theorem 4.2.

For each $\tau \in [L]$, each $\boldsymbol{s} \in \Lambda^\tau$, we regard $\hat{f}_{\mathcal{U}}[\boldsymbol{s}] = L \cdot \hat{f}_{\mathbb{U}_\tau}[\boldsymbol{s}]$ as an estimate of $f_{\mathcal{U}}[\boldsymbol{s}]$. By triangle inequality,

$$\left| L \cdot \hat{f}_{\mathbb{U}_\tau}[\boldsymbol{s}] - f_{\mathcal{U}}[\boldsymbol{s}] \right| \le \left| L \cdot \hat{f}_{\mathbb{U}_\tau}[\boldsymbol{s}] - L \cdot f_{\mathbb{U}_\tau}[\boldsymbol{s}] \right| + \left| L \cdot f_{\mathbb{U}_\tau}[\boldsymbol{s}] - f_{\mathcal{U}}[\boldsymbol{s}] \right|,$$

where $\hat{f}_{\mathbb{U}_\tau}[\boldsymbol{s}]$ is the estimate of $f_{\mathbb{U}_\tau}[\boldsymbol{s}]$ returned by **HadaOracle**. By Corollary 3.2 and Lemma 9.1, it holds with probability at least $1 - \beta'/2$,

$$\left| \hat{f}_{\mathbb{U}_\tau}[\boldsymbol{s}] - f_{\mathbb{U}_\tau}[\boldsymbol{s}] \right| \in O\left(\frac{1}{\varepsilon} \sqrt{|\mathbb{U}_\tau| \ln \frac{1}{\beta'}} \right) \subseteq O\left(\frac{1}{\varepsilon} \sqrt{\frac{n}{L} \ln \frac{1}{\beta'}} \right).$$

By Lemma 9.2, with probability at least $1 - \beta'/2$,

$$|f_{\mathbb{U}_\tau}[\boldsymbol{s}] - f_{\mathcal{U}}[\boldsymbol{s}]/L| \le \|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2 \in O\left(\sqrt{\frac{n}{L} \cdot \ln \frac{L}{\beta'}} \right).$$

By a union bound, with probability at least $1 - \beta'$, it holds that

$$\left| L \cdot \hat{f}_{\mathbb{U}_\tau}[\boldsymbol{s}] - f_{\mathcal{U}}[\boldsymbol{s}] \right| \in O\left(\frac{1}{\varepsilon} \sqrt{n \cdot L \cdot \ln \frac{1}{\beta'}} \right).$$

We finish the proof by substituting $L = 2 \cdot (\log d)/\log n$.

□

In what follows, we need to prove Lemma 9.1 and Lemma 9.2.

First, we prove Lemma 9.1. The lemma holds trivially for **permutation partitioning**, as in such case it holds that $|\mathbb{U}_\tau| = |\mathcal{U}|/L$ for all $\tau \in [L]$. We need to prove the lemma for **independent partitioning**.

9.1.1 Proof of Lemma 9.1 for Independent Partitioning

Let us fix a $\tau \in [L]$, for each $u \in [n]$, define the indicator random variable X_u that equals 1 if $u \in \mathbb{U}_\tau$ and 0 otherwise. Then $|\mathbb{U}_\tau| = \sum_{u \in [n]} X_u$, $\mathbb{E}[|\mathbb{U}_\tau|] = n/L$. By Chernoff bound (Fact 6.1), it holds that

$$\Pr[|\mathbb{U}_\tau| > en/L] \le \left(\frac{e^{e-1}}{e^e} \right)^{n/L} = \frac{1}{e^{n/L}} \,.$$

Recall that $L = 2 \cdot (\log d)/\log n$. Further, in order that the bound $O\left(\frac{1}{\varepsilon}\sqrt{n \cdot \frac{\log d}{\log n} \cdot \ln \frac{1}{\beta'}}\right)$ in Theorem 4.2 to be meaningful, we need the assumption that $n \ge \frac{1}{\varepsilon^2} \cdot L \cdot \ln \frac{1}{\beta'}$. Therefore,

$$\frac{1}{e^{n/L}} \le e^{-\ln(1/\beta')} = \beta' \,.$$

It concludes that $|\mathbb{U}_\tau| \in O(n/L)$ with probability at least $1 - \beta'$.

■

This finishes the proof of Lemma 9.1. Next, we prove Lemma 9.2. We need to prove the lemma for both **permutation partitioning** and **independent partitioning**.

9.1.2 Proof of Lemma 9.2 for permutation partitioning

Consider a fixed $\tau \in [L]$. We have

$$\left\| f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L} \right\|_2 - \mathbb{E}\left[\left\| f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L} \right\|_2 \right] + \mathbb{E}\left[\left\| f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L} \right\|_2 \right] .$$

We will bound each term separately.

Step 1: Bounding $\mathbb{E}\left[\| f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L \|_2\right]$.

By Jensen's inequality,

$$\mathbb{E}\left[\left\| f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L} \right\|_2 \right] \le \sqrt{ \sum_{\boldsymbol{s} \in \Lambda^\tau} \mathbb{E}\left[\left(f_{\mathbb{U}_\tau}[\boldsymbol{s}] - \frac{1}{L} f_{\mathcal{U}}[\boldsymbol{s}] \right)_2^2 \right] } = \sqrt{ \sum_{\boldsymbol{s} \in \Lambda^\tau} \mathbb{V}\mathrm{ar}\left[f_{\mathbb{U}_\tau}[\boldsymbol{s}] \right] } \,.$$

For each user $u \in \mathcal{U}$, define $\boldsymbol{s}^{(u)} \doteq v^{(u)}[1:\tau]$, the prefix of $v^{(u)}$ with length τ. Consider a fixed $\boldsymbol{s} \in \Lambda^\tau$, define $\mathbb{U}[\boldsymbol{s}] \doteq \{u \in \mathcal{U} : \boldsymbol{s}^{(u)} = \boldsymbol{s}\}$ as the set of users holding element $\boldsymbol{s}$. It holds that $|\mathbb{U}[\boldsymbol{s}]| = f_{\mathcal{U}}[\boldsymbol{s}]$. For all $u \in \mathbb{U}[\boldsymbol{s}]$, define the indicator random variable $Z^{(u)}$ to represent the event $u \in \mathbb{U}_\tau$. Then $\Pr[Z^{(u)} = 1] = 1/L$ and $\Pr[Z^{(u)} = 0] = 1 - 1/L$. Then

$$f_{\mathbb{U}_\tau}[\boldsymbol{s}] = \sum_{u \in \mathbb{U}[\boldsymbol{s}]} Z^{(u)} \,,$$

is a sum of $f_{\mathcal{U}}[\boldsymbol{s}]$ dependent random variables with expectation $f_{\mathcal{U}}[\boldsymbol{s}]/L$. For a pair of users $u, u' \in \mathbb{U}[\boldsymbol{s}], u \neq u'$, due to the permutation, if u belongs to $\mathbb{U}_\tau$, it is less likely that u' belongs to $\mathbb{U}_\tau$. In particular,

$$\begin{aligned}
\mathbb{C}\mathrm{ov}\left[Z^{(u)}, Z^{(u')} \right] &= \mathbb{E}\left[Z^{(u)} \cdot Z^{(u')} \right] - \mathbb{E}\left[Z^{(u)} \right] \mathbb{E}\left[Z^{(u')} \right] \\
&= \binom{n-2}{n/L-2} \Big/ \binom{n}{n/L} - \left(\frac{1}{L} \right)^2 = \frac{n/L \cdot (n/L - 1)}{n(n-1)} - \left(\frac{1}{L} \right)^2 \le 0 \,.
\end{aligned}$$

Hence,

$$\mathbb{V}\mathrm{ar}\left[f_{\mathbb{U}_\tau}[\boldsymbol{s}]\right] = \sum_{u \in \mathbb{U}[\boldsymbol{s}]} \mathbb{V}\mathrm{ar}\left[Z^{(u)}\right] + \sum_{u \neq u' \in \mathbb{U}[\boldsymbol{s}]} \mathbb{C}\mathrm{ov}\left[Z^{(u)}, Z^{(u')}\right] \quad \leq \sum_{u \in \mathbb{U}[\boldsymbol{s}]} \mathbb{V}\mathrm{ar}\left[Z^{(u)}\right] = f_{\mathcal{U}}[\boldsymbol{s}] \cdot \frac{1}{L} \cdot \left(1 - \frac{1}{L}\right) \leq \frac{f_{\mathcal{U}}[\boldsymbol{s}]}{L}\,.$$

Therefore,

$$\mathbb{E}\left[\left\| f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L} \right\|_2\right] = \sqrt{\sum_{\boldsymbol{s} \in \Lambda^\tau} \mathbb{V}\mathrm{ar}\left[f_{\mathbb{U}_\tau}[\boldsymbol{s}]\right]} \leq \sqrt{\sum_{\boldsymbol{s} \in \Lambda^\tau} \frac{f_{\mathcal{U}}[\boldsymbol{s}]}{L}} = \sqrt{\frac{n}{L}}\,.$$

Step 2: Bounding $\left\| f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L \right\|_2 - \mathbb{E}\left[\left\| f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L \right\|_2\right]$.

By symmetry, we prove just the case when $\tau = 1$.

Without loss of generality, assume that the n users in $\mathcal{U}$ are indexed by $[n] = \{1, 2, \ldots, n\}$. Let $\mathbf{X} = \{X_1, \ldots, X_n\}$ be a random permutation of $[n]$. As $\left\| f_{\mathbb{U}_1} - f_{\mathcal{U}}/L \right\|_2$ is a function that depends on $\mathbf{X}$, we write it explicitly as $\left\| f_{\mathbb{U}_1} - f_{\mathcal{U}}/L \right\|_2 (X_1, \ldots, X_n)$ or $\left\| f_{\mathbb{U}_1} - f_{\mathcal{U}}/L \right\|_2 (\mathbf{X})$ when necessary. For a possible permutation $\boldsymbol{x} = \{x_1, \ldots, x_n\}$ of $[n]$, we use $\left\| f_{\mathbb{U}_1} - f_{\mathcal{U}}/L \right\|_2 (x_1, \ldots, x_n)$ or $\left\| f_{\mathbb{U}_1} - f_{\mathcal{U}}/L \right\|_2 (\boldsymbol{x})$ to denote the value of $\left\| f_{\mathbb{U}_1} - f_{\mathcal{U}}/L \right\|_2$ when $\mathbf{X} = \boldsymbol{x}$. Let $n_{\text{grp}} = n/L$. For each possible permutation $\boldsymbol{x}$, $\left\| f_{\mathbb{U}_1} - f_{\mathcal{U}}/L \right\|_2 (\boldsymbol{x})$ does not change its value under the change of order of the first n_{grp} and/or last $n - n_{\text{grp}}$ coordinates $\boldsymbol{x}$. Hence, $\left\| f_{\mathbb{U}_1} - f_{\mathcal{U}}/L \right\|_2$ is $(n_{\text{grp}}, n - n_{\text{grp}})$-symmetric (Definition 6.9).

We prove that for each possible permutation $\boldsymbol{x}$, for all $i \in \{1, \ldots, n_{\text{grp}}\}, j \in \{n_{\text{grp}} + 1, \ldots, n\}$, it holds that

$$\left| \left\| f_{\mathbb{U}_1} - \frac{f_{\mathcal{U}}}{L} \right\|_2 (\boldsymbol{x}) - \left\| f_{\mathbb{U}_1} - \frac{f_{\mathcal{U}}}{L} \right\|_2 (\boldsymbol{x}_{i,j}) \right| \leq \sqrt{2}\,, \tag{7}$$

where the permutation $\boldsymbol{x}_{i,j}$ is obtained from $\boldsymbol{x}$ by transposition of its i^{th} and j^{th} coordinates. Then by the McDiarmid Inequality with respect to permutation (Fact 6.10), we have that for all $\eta > 0$,

$$\Pr\left[\left\| f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L} \right\|_2 - \mathbb{E}\left[\left\| f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L} \right\|_2\right] \geq \eta\right] \leq \exp\left(-\frac{2\eta^2}{(n/L) \cdot 2} \left(\frac{n - 1/2}{n - (n/L)}\right) \left(1 - \frac{1}{2\max\{(n/L), n - (n/L)\}}\right)\right).$$

As $L = 2(\log d)/\log n \geq 2$, it holds that $n/L \leq n - n/L$. It follows that

$$\left(\frac{n - 1/2}{n - (n/L)}\right) \left(1 - \frac{1}{2\max\{(n/L), n - (n/L)\}}\right) \geq \left(1 - \frac{1/2}{(n - (n/L))}\right) \geq \frac{1}{2}\,.$$

Substituting η with $\sqrt{2 \cdot (n/L) \cdot \ln(1/\beta')}$, we get

$$\Pr\left[\left\| f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L} \right\|_2 - \mathbb{E}\left[\left\| f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L} \right\|_2\right] \geq \eta\right] \leq \exp\left(-\frac{2\eta^2}{(n/L) \cdot 2} \cdot \frac{1}{2}\right) = \beta'\,.$$

Proof of Inequality (7).

For each $\boldsymbol{s} \in \Lambda$, define $f_{\mathbb{U}_1, \boldsymbol{x}}[\boldsymbol{s}] \doteq \left|\{u \in \mathbb{U}_1 : v^{(u)}[1:1] = \boldsymbol{s}\}\right|$ to be the frequency of $\boldsymbol{s}$ in the set $\{v^{(u)}[1:1] : u \in \mathbb{U}_1\}$, when the random permutation $\mathbf{X} = \boldsymbol{x}$. Let $f_{\mathbb{U}_1, \boldsymbol{x}} \doteq \left(f_{\mathbb{U}_1, \boldsymbol{x}}[\boldsymbol{s}] : \boldsymbol{s} \in \Lambda\right)$ be the corresponding frequency vector when $\mathbf{X} = \boldsymbol{x}$. Similarly, define $f_{\mathbb{U}_1, \boldsymbol{x}_{i,j}}$ to be the frequency vector when $\mathbf{X} = \boldsymbol{x}_{i,j}$.

Let $\boldsymbol{s}^{(x_i)} = v^{(x_i)}[1:1]$ be the prefix of user x_i, and $\boldsymbol{s}^{(x_j)} = v^{(x_j)}[1:1]$ be the prefix of user x_j. When the permutation of $\mathbf{X}$ changes from $\boldsymbol{x}$ to $\boldsymbol{x}_{i,j}$, user x_i is removed from $\mathbb{U}_1$ and user x_j is added into $\mathbb{U}_1$. The frequency of $\boldsymbol{s}^{(x_i)}$ in $\mathbb{U}_1$ decreases by 1, and the frequency of $\boldsymbol{s}^{(x_j)}$ increases by 1.

It holds that

$$f_{\mathbb{U}_1, \boldsymbol{x}_{i,j}}[\boldsymbol{s}] = \begin{cases} f_{\mathbb{U}_1, \boldsymbol{x}}[\boldsymbol{s}], & \forall \boldsymbol{s} \in \Lambda \setminus \{\boldsymbol{s}^{(x_i)}, \boldsymbol{s}^{(x_j)}\}\,, \\ f_{\mathbb{U}_1, \boldsymbol{x}}[\boldsymbol{s}] - 1, & \boldsymbol{s} = \boldsymbol{s}^{(x_i)}\,, \\ f_{\mathbb{U}_1, \boldsymbol{x}}[\boldsymbol{s}] + 1, & \boldsymbol{s} = \boldsymbol{s}^{(x_j)}\,. \end{cases}$$

If we view $f_{\mathbb{U}_1,\boldsymbol{x}} - f_{\mathcal{U}}/L$ and $f_{\mathbb{U}_1,\boldsymbol{x}_{i,j}} - f_{\mathcal{U}}/L$ as $|\Lambda|$-dimensional vectors, it holds that

$$\left(f_{\mathbb{U}_1,\boldsymbol{x}} - \frac{f_{\mathcal{U}}}{L}\right) - \left(f_{\mathbb{U}_1,\boldsymbol{x}_{i,j}} - \frac{f_{\mathcal{U}}}{L}\right) = -\boldsymbol{e}_{\boldsymbol{s}^{(x_i)}} + \boldsymbol{e}_{\boldsymbol{s}^{(x_j)}}\ ,$$

where $\boldsymbol{e}_{\boldsymbol{s}^{(x_i)}}$ and $\boldsymbol{e}_{\boldsymbol{s}^{(x_j)}}$ are the $\boldsymbol{s}^{(x_i)}$-th and the $\boldsymbol{s}^{(x_j)}$-th standard basis vectors in $\mathbb{R}^{|\Lambda|}$ respectively. By the triangle inequality,

$$\left\|f_{\mathbb{U}_1,\boldsymbol{x}} - \frac{f_{\mathcal{U}}}{L}\right\|_2 - \left\|f_{\mathbb{U}_1,\boldsymbol{x}_{i,j}} - \frac{f_{\mathcal{U}}}{L}\right\|_2 \le \| - \boldsymbol{e}_{\boldsymbol{s}^{(x_i)}} + \boldsymbol{e}_{\boldsymbol{s}^{(x_j)}}\|_2 = \sqrt{2}\,.$$

■

This finishes the proof of Lemma 9.2 for **permutation partitioning**. Next, we prove Lemma 9.2 for **independent partitioning**.

9.1.3 Proof of Lemma 9.2 for independent partitioning

Without loss of generality, we prove this lemma for a fixed $\tau \in [L]$. To simplify the notation, for each user $u \in \mathcal{U}$, we write $\boldsymbol{s}^{(u)} \doteq v^{(u)}[1:\tau]$ as the prefix of $v^{(u)}$ with length τ. For each $u \in \mathcal{U}$, define the indicator random variable X_u for the event $u \in \mathbb{U}_\tau$. Let $\mathbf{X}$ be shorthand for $\{X_1, \dots, X_n\}$. As $\|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2$ is a function that depends on $\mathbf{X}$, we write it explicitly as $\|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2\,(X_1, \dots, X_n)$ or $\|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2\,(\mathbf{X})$ when necessary. For a sequence of values $\boldsymbol{x} = \{x_1, \dots, x_n\} \in \{0,1\}^n$, we use $\|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2\,(x_1, \dots, x_n)$ or $\|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2\,(\boldsymbol{x})$ to denote the value of $\|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2\,(\mathbf{X})$, when $\mathbf{X} = \boldsymbol{x}$. Since

$$\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2 = \left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2 - \mathbb{E}\left[\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2\right] + \mathbb{E}\left[\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2\right]\ ,$$

we can bound $\|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2 - \mathbb{E}\left[\|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2\right]$ and $\mathbb{E}\left[\|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2\right]$ separately.

Step 1: Bounding $\mathbb{E}\left[\|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2\right]$.

First, similar to the proof for Lemma 8.2, we have that

$$\mathbb{E}\left[\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2^2\right] = \sum_{\boldsymbol{s}\in\Lambda^\tau} \mathbb{E}\left[\left(f_{\mathbb{U}_\tau}[\boldsymbol{s}] - \frac{1}{L} f_{\mathcal{U}}[\boldsymbol{s}]\right)^2\right] = \sum_{\boldsymbol{s}\in\Lambda^\tau} \mathbb{V}\mathrm{ar}\left[f_{\mathbb{U}_\tau}[\boldsymbol{s}]\right] \le \sum_{\boldsymbol{s}\in\Lambda^\tau} \frac{1}{L} f_{\mathcal{U}}[\boldsymbol{s}] = \frac{n}{L}\,.$$

Hence, by Jensen's inequality, $\mathbb{E}\left[\|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2\right] \le \sqrt{\mathbb{E}\left[\|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2^2\right]} \in O\left(\sqrt{n/L}\right)$.

Step 2: Bounding $\|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2 - \mathbb{E}\left[\|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2\right]$.

Bounding this is more nuanced than the equivalent term in the proof of Lemma 8.2. We could show that $\|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2$ satisfies the Lipschitz condition (Definition 6.7) with bound 1 and then apply McDiarmid's Inequality (Fact 6.8). But this will give us an inferior bound of $O\left(\sqrt{n\ln(1/\beta')}\right)$. The Lipschitz condition states that for each $u \in \mathcal{U}$, if the value of X_u changes, it affects $\|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2$ by at most 1. This completely ignores the variance of $\|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2$.

We apply a martingale concentration inequality that incorporates variances (Fact 6.6). First, we introduce a dummy variable $X_0 \equiv 0$. For each $0 \le i \le n$, let $\mathbf{S}_i$ be shorthand for $(X_0, \dots, X_i)$. Define

$$Y_i \doteq \mathbb{E}\left[\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2 \,\middle|\, \mathbf{S}_i\right].$$

Clearly Y_i is a function of $X_0, \dots, X_i$ and $\mathbb{E}[|Y_i|] \le \infty$. Moreover,

$$\mathbb{E}[Y_{i+1} \mid \mathbf{S}_i] = \mathbb{E}\left[\mathbb{E}\left[\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2 \,\middle|\, \mathbf{S}_i, X_{i+1}\right] \,\middle|\, \mathbf{S}_i\right] = \mathbb{E}\left[\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2 \,\middle|\, \mathbf{S}_i\right] = Y_i.$$

The sequence $Y_0, \ldots, Y_n$ satisfies all conditions specified in Definition 6.4 and is a martingale. By definition, $Y_n = \mathbb{E}\left[\|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2 \,|\, \mathbf{S}_n\right] = \|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2$ as, once the values of $X_1, \ldots, X_n$ are determined, so is $\|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2$. And we have $Y_0 = \mathbb{E}\left[\|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2 \,|\, X_0\right] = \mathbb{E}\left[\|f_{\mathbb{U}_\tau} - f_{\mathcal{U}}/L\|_2\right]$, as $X_0 \equiv 0$.

Next, we show that for all $i \in [n]$,

1. $|Y_i - Y_{i-1}| \le 1$; and
2. $\mathbb{V}\mathrm{ar}[Y_i \mid X_0, \ldots, X_{i-1}] \le 1/L$.

Then by the concentration inequality (Fact 6.6), it holds that for all $\eta > 0$,

$$\Pr[Y_n - Y_0 \ge \eta] \le \exp\left(-\frac{\eta^2}{2\left(n/L + \eta/3\right)}\right).$$

We want to find an η, s.t., $\exp\left(-\frac{\eta^2}{2(n/L+\eta/3)}\right) = \beta'$. This leads to an equation

$$\eta^2 - \frac{2 \cdot \ln(1/\beta')}{3}\eta - \frac{2 \cdot n \cdot \ln(1/\beta')}{L} = 0\,,$$

whose solution gives

$$\eta = \frac{\ln(1/\beta')}{3} + \frac{1}{2}\sqrt{\frac{2^2 \cdot \ln^2(1/\beta')}{3^2} + \frac{8 \cdot n \cdot \ln(1/\beta')}{L}} \le \sqrt{\frac{2 \cdot n \cdot \ln(1/\beta')}{L}} + \frac{2 \cdot \ln(1/\beta')}{3}\,.$$

We conclude that, with probability at least $1 - \beta'$, it holds that

$$\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2 - \mathbb{E}\left[\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2\right] \le \sqrt{2 \cdot \frac{n}{L} \cdot \ln\frac{1}{\beta'}} + \frac{2 \cdot \ln(1/\beta')}{3} \le \left(\sqrt{2} + \frac{2}{3}\right) \cdot \sqrt{\frac{n}{L} \cdot \ln\frac{1}{\beta'}}\,.$$

Proving that for all $i \in [n], |Y_i - Y_{i-1}| \le 1$.

For all sequences $\boldsymbol{x}_{i-1} = (0, x_1, \ldots, x_{i-1}) \in \{0,1\}^i$ and for all $x_i \in \{0,1\}$, we prove that

$$\left|\mathbb{E}\left[\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2 \middle| \mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}, X_i = x_i\right] - \mathbb{E}\left[\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2 \middle| \mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}\right]\right| \le 1\,.$$

Note that

$$\mathbb{E}\left[\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2 \middle| \mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}\right] = \mathop{\mathbb{E}}_{x \in \{0,1\}}\left[\mathbb{E}\left[\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2 \middle| \mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}, X_i = x\right]\right].$$

Define

$$\gamma \doteq \left|\mathbb{E}\left[\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2 \middle| \mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}, X_i = 0\right] - \mathbb{E}\left[\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2 \middle| \mathbf{S}_{i-1} = \boldsymbol{x}_{i-1}, X_i = 1\right]\right|.$$

It suffices to prove that $\gamma \le 1$. Let $\boldsymbol{x}_i^+ = (x_{i+1}, \ldots, x_n)$ denote a possible sequence in $\{0,1\}^{n-i}$. For $x_i \in \{0,1\}$, we write

$$\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2 (\boldsymbol{x}_{i-1}, x_i, \boldsymbol{x}_i^+) \qquad \text{for} \qquad \left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2 (x_0, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n).$$

Let $\mathbf{S}_i^+$ be shorthand for $\{X_{i+1}, \ldots, X_n\}$. Since $\mathbf{S}_i^+$ is independent of $\mathbf{S}_i$, we have

$$\gamma = \left|\sum_{\boldsymbol{x}_i^+ \in \{0,1\}^{n-i}} \Pr[\mathbf{S}_i^+ = \boldsymbol{x}_i^+] \cdot \left(\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2 (\boldsymbol{x}_{i-1}, 0, \boldsymbol{x}_i^+) - \left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2 (\boldsymbol{x}_{i-1}, 1, \boldsymbol{x}_i^+)\right)\right|.$$

Consider a fixed $\boldsymbol{x}_i^+ \in \{0,1\}^{n-i}$. Let $\boldsymbol{x} = (\boldsymbol{x}_{i-1}, 0, \boldsymbol{x}_i^+)$ and $\boldsymbol{x}' = (\boldsymbol{x}_{i-1}, 1, \boldsymbol{x}_i^+)$. Let $f_{\mathbb{U}_\tau, \boldsymbol{x}}$ and $f_{\mathbb{U}_\tau, \boldsymbol{x}'}$ be the frequency vectors defined on $\mathbb{U}_\tau$, when $\mathbf{X} = \boldsymbol{x}$ and $\mathbf{X} = \boldsymbol{x}'$, respectively. Let $\boldsymbol{s}^{(i)} = v^{(i)}[1:\tau]$ be the prefix with length τ of user i. For $\boldsymbol{s} \in \Lambda^\tau \setminus \{\boldsymbol{s}^{(i)}\}$, it holds that $f_{\mathbb{U}_\tau, \boldsymbol{x}}[\boldsymbol{s}] = f_{\mathbb{U}_\tau, \boldsymbol{x}'}[\boldsymbol{s}]$. Further, $f_{\mathbb{U}_\tau, \boldsymbol{x}}[\boldsymbol{s}^{(i)}] + 1 = f_{\mathbb{U}_\tau, \boldsymbol{x}'}[\boldsymbol{s}^{(i)}]$.

If we view $f_{\mathbb{U}_\tau,\boldsymbol{x}} - f_{\mathcal{U}}/L$ and $f_{\mathbb{U}_\tau,\boldsymbol{x}'} - f_{\mathcal{U}}/L$ as $|\Lambda^\tau|$-dimensional vectors, it holds that

$$\left(f_{\mathbb{U}_\tau,\boldsymbol{x}} - \frac{f_{\mathcal{U}}}{L}\right) - \left(f_{\mathbb{U}_\tau,\boldsymbol{x}'} - \frac{f_{\mathcal{U}}}{L}\right) = -\boldsymbol{e}_{\boldsymbol{s}^{(i)}}\,,$$

where $\boldsymbol{e}_{\boldsymbol{s}^{(i)}}$ is the $\boldsymbol{s}^{(i)}$-th standard basis vector. By the triangle inequality,

$$\left\|f_{\mathbb{U}_\tau,\boldsymbol{x}} - \frac{f_{\mathcal{U}}}{L}\right\|_2 - \left\|f_{\mathbb{U}_\tau,\boldsymbol{x}'} - \frac{f_{\mathcal{U}}}{L}\right\|_2 \le \|\boldsymbol{e}_{\boldsymbol{s}^{(i)}}\|_2 = 1\,.$$

Proving that for all $i \in [n], \mathbb{V}\mathbf{ar}[Y_i \mid X_0, \ldots, X_{i-1}] \le 1/L$.

For each sequence $\boldsymbol{x}_{i-1}$,

$$\mathbb{E}\left[\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2 \,\middle|\, \mathbf{X}_{i-1} = \boldsymbol{x}_{i-1}\right] = \begin{cases} \mathbb{E}\left[\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2 \,\middle|\, \mathbf{X}_{i-1} = \boldsymbol{x}_{i-1}, X_i = 1\right], & \text{w.p. } 1/L\,, \\ \mathbb{E}\left[\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2 \,\middle|\, \mathbf{X}_{i-1} = \boldsymbol{x}_{i-1}, X_i = 0\right], & \text{w.p. } 1 - 1/L\,. \end{cases}$$

We have also proven that

$$\left|\mathbb{E}\left[\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2 \,\middle|\, \mathbf{X}_{i-1} = \boldsymbol{x}_{i-1}, X_i = 0\right] - \mathbb{E}\left[\left\|f_{\mathbb{U}_\tau} - \frac{f_{\mathcal{U}}}{L}\right\|_2 \,\middle|\, \mathbf{X}_{i-1} = \boldsymbol{x}_{i-1}, X_i = 1\right]\right| \le 1.$$

It follows that $\mathbb{V}\mathrm{ar}\left[Y_i \mid \mathbf{X}_{i-1} = \boldsymbol{x}_{i-1}\right] \le 1/L \cdot (1 - 1/L) \le 1/L$.

■

This finishes the proof of Theorem 4.2. Next, we prove Theorem 4.3.

9.2 Theorem 4.3

Theorem 4.3. *Let $\lambda \doteq 3 \cdot \lambda'$. With probability at least $1 - \beta$, it is guaranteed that for each $\tau \in [L]$, and each $\boldsymbol{s} \in \Lambda^\tau$: (1) if $f_{\mathcal{U}}[\boldsymbol{s}] \ge \lambda$, then $\boldsymbol{s} \in \mathcal{P}_\tau$; (2) and for each $\boldsymbol{s} \in \mathcal{P}_\tau$, the frequency estimate $\hat{f}_{\mathcal{U}}[\boldsymbol{s}]$ satisfies $|\hat{f}_{\mathcal{U}}[\boldsymbol{s}] - f_{\mathcal{U}}[\boldsymbol{s}]| \le \lambda'$. Constructing the $\mathcal{P}_\tau$ for all $\tau \in [L]$ has $\tilde{O}(n)$ running time and $\tilde{O}(\sqrt{n})$ memory usage.*

Proof. We focus on the estimation errors of prefixes from a fixed set.

Definition 4.4 Define $\Gamma_0 \doteq \{\perp\}$ to be the set of the empty string, and for $\tau \in [L]$, $\Gamma_\tau \doteq \{\boldsymbol{s} \in \Lambda^\tau : f_{\mathcal{U}}[\boldsymbol{s}] \ge \lambda'\}$, the set of prefixes of length τ whose frequency is at least λ'. For $\tau < L$, the child set of Γ_τ is defined as $\Gamma_\tau \times \Lambda \doteq \{\boldsymbol{s} = \boldsymbol{s}_1 \circ \boldsymbol{s}_2 : \boldsymbol{s}_1 \in \Gamma_\tau, \boldsymbol{s}_2 \in \Lambda\}$, where $\boldsymbol{s}_1 \circ \boldsymbol{s}_2$ is the concatenation of $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$. The *candidate* set is defined as $\Gamma \doteq \cup_{0 \le \tau < L} (\Gamma_\tau \times \Lambda)$.

Note that for each $\tau \in [L]$, we have $|\Gamma_\tau| \le n/\lambda' \le \sqrt{n}$. Hence, $|\Gamma| = \sum_{0 \le \tau < L} |\Gamma_\tau \times \Lambda| \le L\sqrt{n} \cdot \sqrt{n} \in \tilde{O}(n)$. By applying Theorem 4.2 with $\beta' = \beta/(nL)$ and the union bound over all $\boldsymbol{s} \in \Gamma$, we have:

Corollary 4.5 *There exists some constant C_λ, such that with probability at least $1-\beta$, it holds that $\max_{\boldsymbol{s}\in\Gamma} |\hat{f}_{\mathcal{U}}[\boldsymbol{s}] - f_{\mathcal{U}}[\boldsymbol{s}]| \le \lambda'$, where*

$$\lambda' = (C_\lambda/\varepsilon)\sqrt{n \cdot (\log d) \cdot (\ln(n/\beta))/\ln n}\,.$$

Lemma 9.3. *Suppose that all strings in Γ having estimation error λ', i.e., for each $\boldsymbol{s} \in \Gamma$, it holds that $|\hat{f}_{\mathcal{U}}[\boldsymbol{s}] - f_{\mathcal{U}}[\boldsymbol{s}]| \le \lambda'$. It is follows that for each $\tau \in [L]$, and for each $\boldsymbol{s} \in \Lambda^\tau$: i) if $f_{\mathcal{U}}[\boldsymbol{s}] \ge 3 \cdot \lambda'$, then $\boldsymbol{s} \in \mathcal{P}_\tau$; ii) if $f_{\mathcal{U}}[\boldsymbol{s}] < \lambda'$, then $\boldsymbol{s} \notin \mathcal{P}_\tau$.*

The proof of the lemma is by induction, and is rather technical. We defer it to the end of the proof. By now, we finish the proof of the theorem based on this lemma.

Conditioned all strings in Γ having estimation error λ', via the lemma, we see that for each $\tau \in [L]$, and each $\boldsymbol{s} \in \Lambda^\tau$, if $f_{\mathcal{U}}[\boldsymbol{s}] \geq \lambda = 3\lambda'$, then $\boldsymbol{s}$ is guaranteed to be added to $\mathcal{P}_\tau$.

Moreover, the second condition of the lemma implies that for each $\tau \in [L]$, and each $\boldsymbol{s} \in \mathcal{P}_\tau$, it holds that $f_{\mathcal{U}}[\boldsymbol{s}] \geq \lambda'$. By the definition of Γ_τ, we have $\boldsymbol{s} \in \Gamma_\tau$. Hence $\mathcal{P}_\tau \subset \Gamma_\tau$. By the assumption that all string in Γ have estimation error bounded by λ', we the see that strings in $\mathcal{P}_\tau$ have estimation errors bounded by λ'. Lastly, note that for $0 \leq \tau < L$, the *modified search strategy* constructs $\mathcal{P}_{\tau+1}$ based on $\mathcal{P}_\tau$, by checking strings in $\mathcal{P}_\tau \times \Lambda$. It follows that all such strings belong to $\Gamma_\tau \times \Lambda \subset \Gamma$. Therefore, the *modified search strategy* only inspects the frequencies of the strings from Γ, in order to construct the $\mathcal{P}_\tau, \tau \in [L]$.

To analyze the running time and memory usage, observe that the *modified search strategy* invokes L frequency oracles. By Theorem 3.2, they have total construction time $\tilde{O}(n)$ and memory usage $\tilde{O}(\sqrt{n})$. Since each frequency query takes $\tilde{O}(1)$ time, and all strings queried belong to Γ, the total query time is bounded by $|\Gamma| \in \tilde{O}(n)$, which finishes the proof. □

Proof for Lemma 9.3. Suppose that all strings in Γ having estimation error λ', i.e., for each $\boldsymbol{s} \in \Gamma$, it holds that $|\hat{f}_{\mathcal{U}}[\boldsymbol{s}] - f_{\mathcal{U}}[\boldsymbol{s}]| \leq \lambda'$. We prove the claims by induction.

When $\tau = 1$, $\mathcal{P}_{\tau-1} = \Gamma_{\tau-1} = \{\perp\}$. Then $\mathcal{P}_0 \times \Lambda = \Gamma_0 \times \Lambda = \Lambda$. For all $\boldsymbol{s} \in \Lambda$, if $f_{\mathcal{U}}[\boldsymbol{s}] \geq 3 \cdot \lambda'$, then it holds that

$$\hat{f}_{\mathcal{U}}[\boldsymbol{s}] \geq f_{\mathcal{U}}[\boldsymbol{s}] - \lambda' \geq 2\lambda' .$$

According to the definition of $\mathcal{P}_1$, the element $\boldsymbol{s}$ belongs to $\mathcal{P}_1$. On the other hand, if $f_{\mathcal{U}}[\boldsymbol{s}] < \lambda'$, then

$$\hat{f}_{\mathcal{U}}[\boldsymbol{s}] \leq f_{\mathcal{U}}[\boldsymbol{s}] + \lambda' < 2\lambda' .$$

It is guaranteed that $\boldsymbol{s} \notin \mathcal{P}_1$. Consequently, for all $\boldsymbol{s} \in \mathcal{P}_1$, we have $f_{\mathcal{U}}[\boldsymbol{s}] \geq \lambda'$, which implies that $\mathcal{P}_1 \subseteq \Gamma_1$.

Let $\tau > 1$ and assume that the claims holds for $\tau - 1$. For all $\boldsymbol{s} \in \Lambda^\tau$, let $\boldsymbol{s}[1 : \tau - 1] \in \Lambda^{\tau-1}$ be the prefix of $\boldsymbol{s}$ of length $\tau - 1$. If $f_{\mathcal{U}}[\boldsymbol{s}] \geq 3 \cdot \lambda'$, then it holds that

$$f_{\mathcal{U}}[\boldsymbol{s}[1 : \tau - 1]] \geq f_{\mathcal{U}}[\boldsymbol{s}] \geq 3\lambda' .$$

By the induction hypothesis, $\boldsymbol{s}[1 : \tau - 1] \in \mathcal{P}_{\tau-1} \subseteq \Gamma_{\tau-1}$. Hence, $\boldsymbol{s} \in \mathcal{P}_{\tau-1} \times \Lambda \subseteq \Gamma_{\tau-1} \times \Lambda$, and

$$\hat{f}_{\mathcal{U}}[\boldsymbol{s}] \geq f_{\mathcal{U}}[\boldsymbol{s}] - \lambda' \geq 2\lambda' .$$

According to the definition of $\mathcal{P}_\tau$, the element $\boldsymbol{s}$ belongs to $\mathcal{P}_\tau$.

On the other hand, if $f_{\mathcal{U}}[\boldsymbol{s}] < \lambda'$, there are two possible cases. First, if $\boldsymbol{s}[1 : \tau - 1] \notin \mathcal{P}_{\tau-1}$, then by the definition of $\mathcal{P}_\tau$, it is guaranteed that $\boldsymbol{s} \notin \mathcal{P}_\tau$. Second, if $\boldsymbol{s}[1 : \tau - 1] \in \mathcal{P}_{\tau-1}$, by the inductive hypothesis that $\mathcal{P}_{\tau-1} \subset \Gamma_{\tau-1}$, we have $\boldsymbol{s} \subset \mathcal{P}_{\tau-1} \times \Lambda \subset \Gamma_{\tau-1} \times \Lambda$. Then

$$\hat{f}_{\mathcal{U}}[\boldsymbol{s}] \leq f_{\mathcal{U}}[\boldsymbol{s}] + \lambda' < 2\lambda' .$$

It is guaranteed that $\boldsymbol{s} \notin \mathcal{P}_\tau$. Consequently, for all $\boldsymbol{s} \in \mathcal{P}_\tau$, we have $f_{\mathcal{U}}[\boldsymbol{s}] \geq \lambda'$, which implies that $\mathcal{P}_\tau \subseteq \Gamma_\tau$.

□