Frequency Estimation in the Shuffle Model with Almost a Single Message

Qiyao Luo
qluoak@cse.ust.hk
HKUST
Hong Kong, China

Yilei Wang
fengmi.wyl@alibaba-inc.com
Alibaba Group
China

Ke Yi
yike@cse.ust.hk
HKUST
Hong Kong, China

ABSTRACT

We present a protocol in the shuffle model of differential privacy (DP) for the *frequency estimation* problem that achieves error $\omega(1) \cdot O(\log n)$, almost matching the central-DP accuracy, with $1 + o(1)$ messages per user. This exhibits a sharp transition phenomenon, as there is a lower bound of $\Omega(n^{1/4})$ if each user is allowed to send only one message. Previously, such a result is only known when the domain size B is $o(n)$. For a large domain, we also need an efficient method to identify the *heavy hitters* (i.e., elements that are frequent enough). For this purpose, we design a shuffle-DP protocol that uses $o(1)$ messages per user and can identify all heavy hitters in time polylogarithmic in B. Finally, by combining our frequency estimation and the heavy hitter detection protocols, we show how to solve the B-dimensional *1-sparse vector summation* problem in the high-dimensional setting $B = \Omega(n)$, achieving the optimal central-DP MSE $\tilde{O}(n)$ with $1 + o(1)$ messages per user. In addition to error and message number, our protocols improve in terms of message size and running time as well. They are also very easy to implement. The experimental results demonstrate order-of-magnitude improvement over prior work.

CCS CONCEPTS

• **Security and privacy → Privacy-preserving protocols**.

KEYWORDS

Differential privacy, frequency estimation, heavy hitter, sparse vector summation

ACM Reference Format:
Qiyao Luo, Yilei Wang, and Ke Yi. 2022. Frequency Estimation in the Shuffle Model with Almost a Single Message. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security (CCS '22), November 7–11, 2022, Los Angeles, CA, USA.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3548606.3560608

CCS '22, November 7–11, 2022, Los Angeles, CA, USA.

ACM ISBN 978-1-4503-9450-5/22/11…$15.00
https://doi.org/10.1145/3548606.3560608

1 INTRODUCTION

1.1 Central-DP, local-DP, and shuffle-DP

A randomized mechanism $\mathcal{M} : \mathcal{X}^n \to \mathcal{Y}$ is (ε, δ)-*differentially private (DP)* for some privacy parameters $\varepsilon > 0$ and $0 \le \delta < n^{-\Omega(1)}$, if for any two neighboring datasets $\boldsymbol{D} \sim \boldsymbol{D}'$ (i.e., $\boldsymbol{D}$ and $\boldsymbol{D}'$ differ by one element), and any set of outputs $Y \subseteq \mathcal{Y}$,

$$\Pr[\mathcal{M}(\boldsymbol{D}) \in Y] \le e^{\varepsilon} \cdot \Pr[\mathcal{M}(\boldsymbol{D}') \in Y] + \delta. \tag{1}$$

Depending on how $\mathcal{M}(\boldsymbol{D})$ is defined, we arrive at different DP models. In *central-DP*, there is a trusted curator who has direct access to the entire dataset $\boldsymbol{D} = (x_1, \ldots, x_n)$ and $\mathcal{M}(\boldsymbol{D})$ is the privatized estimate of the desired function. In *local-DP*, each user i locally privatizes their own datum x_i using a randomizer $\mathcal{R}$, and sends $\mathcal{R}(x_i)$ to an untrusted analyzer $\mathcal{A}$. The DP requirement (1) shall hold on these messages, i.e., $\mathcal{M}(\boldsymbol{D}) = (\mathcal{R}(x_1), \ldots, \mathcal{R}(x_n))$. Local-DP provides an alternative approach to *secure multi-party computation (MPC)* with the following advantages:

- Simplicity: A local-DP protocol, by definition, uses only one round of one-way communications, as opposed to most MPC protocols that require multiple rounds of two-way interactions.
- Efficiency: Local-DP protocols are usually much more efficient than their MPC counterparts in terms of both communication and computation. Its communication cost is often $\tilde{O}(n)$, as opposed to at least $\Omega(n^2)$ even for the most basic MPC protocol (assuming a constant fraction of colluding parties), such as bit AND [18]. Furthermore, most local-DP protocols use simple arithmetic operations without any expensive cryptography.

These advantages make local-DP a promising direction for large-scale aggregation problems over private data, which has already been adopted by Apple [1], Google [24], and Microsoft [32]. But it also has clear weaknesses:

(1) Weaker privacy guarantee: An MPC transcript reveals information about the private input with negligible probability, but in local-DP, ε is usually set to a constant[1] [1, 24].
(2) Low accuracy: While an MPC protocol can compute a given functionality with 100% accuracy, local-DP protocols incur errors at least $\Omega(\sqrt{n})$ even for the most basic problems (e.g., bit counting) and this is inherent [12, 14, 37].

The first weakness can often be mitigated in the large-scale setting, as concerned users may opt out, which would not affect the aggregation accuracy too much as long as enough users participate.

[1] Thus, in the introduction, we state all results for a constant ε, and assume $\Theta(\log \frac{1}{\delta}) = \Theta(\log \frac{1}{\beta}) = \Theta(\log B) = \Theta(\log n)$ to simplify the bounds. The $\tilde{O}$ notation further suppresses these polylogarithmic factors.

In addition, technically speaking, the privacy guarantees of MPC and DP are incomparable: An MPC protocol satisfies (1) with $\varepsilon = 0$ over any two $\boldsymbol{D}, \boldsymbol{D}'$ having the same function output but not necessarily neighbors. This offers stronger protection on the transcript, but it does not prevent the output from revealing sensitive information. Thus, an MPC protocol is often combined with differential privacy to provide all-round protection, i.e., a central-DP mechanism for the function is implemented inside the MPC [28, 31, 34]. This makes the strong privacy guarantee of MPC a bit "excessive", since the overall privacy guarantee of the whole system is decided by the weakest link, which is differential privacy.

However, the second weakness is still a major concern for local-DP. In fact, it is due to this reason that the above MPC + central-DP approach is still popular, since a central-DP mechanism has error $\tilde{O}(1)$ for many problems.

Instead of MPC + central-DP, which has good accuracy but high costs, a different approach has been suggested, by assuming that the $\mathcal{R}(x_i)$'s are sent to the analyzer anonymously [30]. Combined with differential privacy, this results in the *shuffle-DP* model [4, 5, 10, 14, 15, 25, 27]. More precisely, shuffle-DP applies the requirement (1) on the multiset $\mathcal{M}(\boldsymbol{D}) = \{\mathcal{R}(x_1), \ldots, \mathcal{R}(x_n)\}$, or equivalently, the messages are given to the analyzer after a random shuffle. Shuffle-DP keeps the simplicity and efficiency of local-DP, modulo the cost of the shuffler, while possibly offering significant improvement in terms of accuracy. For the bit counting problem, a simple randomized response protocol, in which each user sends a random bit with probability $\tilde{O}(1/n)$, otherwise the true input x_i, can already achieve $\tilde{O}(1)$ error in shuffle-DP, matching the error in central-DP up to polylogarithmic factors.

Single-message shuffle-DP. In the randomized response protocol above, each user only sends one message (actually, just one bit). Moving beyond bit counting, people realized that the single-message shuffle-DP model has its limitation. Generalizing bit counting to the *real summation* problem, i.e., each $x_i \in [0, 1]$ and we wish to estimate $\sum_i x_i$, we encounter a lower bound of $\Omega(n^{1/6})$ [4]. For the frequency estimation problem, which is the focus of this paper, there is a lower bound of $\Omega(n^{1/4})$ [25]. On the other hand, central-DP can still achieve error $\tilde{O}(1)$ for both problems.

Multi-message shuffle-DP. In view of the lower bounds above, it is natural to study shuffle-DP protocols where each user sends multiple messages. For real summation, the protocol in [30] achieves $O(1)$ error by sending $O(\log n)$ messages per user. This was later improved to $O(1)$ messages [5]. Recently, Ghazi et al. [27] have designed a simple and elegant protocol that further reduces the (expected) number of messages per user to $1 + o(1)$, and reduces the constant in the $O(1)$ error bound to almost match the error achievable in central-DP. The reduction in the message number is important both theoretically and practically: Theoretically, it exhibits a sharp transition phenomenon that $o(1)$ extra messages can bring a polynomial improvement in the error; practically, reducing the message number is important since each message has to be anonymized by the shuffler, which could be the dominating cost in a shuffle-DP system, depending on the implementation of the shuffler and its trust assumptions.

1.2 Frequency Estimation

In the frequency estimation problem, each of the n users holds an element from a domain $[B] := \{0, \ldots, B-1\}$ and the goal is to estimate the number of users holding element x for any $x \in [B]$. Let $\boldsymbol{D} = (x_1, \ldots, x_n) \in [B]^n$ be the input, and let $g_x = \sum_{i=1}^n \mathbb{I}[x_i = x]$ be the frequency of x in $\boldsymbol{D}$. A frequency estimation protocol $P : [B]^n \to \mathbb{N}^B$ returns an estimate of g_x for each $x \in [B]$. We are interested in the maximum estimation error, i.e., for a failure probability β, we say that P has error α if

$$\Pr\left[\max_{x \in [B]} |P(\boldsymbol{D})_x - g_x| > \alpha\right] < \beta.$$

When $B = 2$, the problem degenerates into bit counting. But for the frequency estimation problem, we are more interested in the large-domain case $B \gg n$. For example, when estimating the popularity of hashtags and websites without any *a priori* restrictions on the possible hashtags/websites, the domain would be all strings (up to a certain length) and all IP addresses. For large B, $P(\boldsymbol{D})$ shall be returned implicitly, i.e., it is a data structure from which $P(\boldsymbol{D})_x$ can be extracted for any given query x.

Ghazi et al. [25] prove a lower bound of $\Omega(n^{1/4})$ if each user is only allowed a single message; they also present a protocol that uses $O(\log n)$ messages to achieve $O(\log n)$ error, matching the optimal error in central-DP. In this paper, we present a new shuffle-DP frequency estimation protocol that achieves (i) $O(\log n)$ error with $O(1)$ messages; or (ii) $\omega(1) \cdot O(\log n)$ error with $1 + o(1)$ messages, where $\omega(1)$ denotes any super-constant function. Thus, similar to the real summation problem, we see a sharp transition phenomenon for the frequency estimation problem as well.

In addition to error and message complexity, our protocol also reduces each message size from $O(\log^2 n)$ bits [25] to $O(\log n)$ bits, and reduce the query time from $O(n \log^3 n)$ [25] to $O(n)$. It is also easy to implement, requires only private randomness, and defends against any constant fraction of malicious users (known as *robust shuffle-DP*).

1.3 Heavy Hitter Detection

For a large domain, estimating the frequency of each element one by one is impractical. Instead, it should be equipped with a *heavy hitter detection* technique to identify the elements that are frequent enough. For a threshold $0 < \phi < 1$, the set of *heavy hitters* are $\{x \mid g_x \geq \phi n, x \in [B]\}$. Note that there are at most $1/\phi$ heavy hitters, thus the goal of heavy hitter detection is to find all of them (their identifiers) in time that depends on $1/\phi$. Afterwards, one can query a frequency estimation data structure to obtain their frequencies and remove the false positives, subject to an error of $\tilde{O}(1)$.

The heavy hitter detection shuffle-DP protocol in [25] works for any $\phi = \Omega(\log^2 n/n)$, sends $O(\log^4 n)$ messages per user, and the detection time is $O(n \log^6 n/\phi)$. In this paper, we present an improved technique to detect all heavy hitters for any $\phi = \Omega(\log^2 n/n)$ using $O(\log^2 n/\phi n)$ messages and the detection time is $O(\log^2 n/\phi^2)$, both of which are better than [25] for all $\phi = \Omega(\log^2 n/n)$, and the improvement is more significant for larger ϕ. In particular, for $\phi = \omega(\log^2 n/n)$, the message number is $o(1)$. Combined with our $1+o(1)$ frequency estimation protocol, we obtain a $1+o(1)$ protocol

to estimate the frequencies of all elements with error $\tilde{O}(1)$ in time $O(n^2)$ for any $B = n^{O(1)}$, where we simply set the frequencies of the light hitters to 0.

1.4 1-Sparse Vector Summation

Finally, we consider the *1-sparse vector summation* problem [27]. The i-th user holds an element $x_i \in [B]$ together with a weight $w_i \in [0, 1]$, and the goal is to estimate $g_x = \sum_{i=1}^{n} w_i \cdot \mathbb{I}[x_i = x]$ for all $x \in [B]$. In other words, each user holds a 1-sparse B-dimensional vector $\boldsymbol{v_i} \in [0, 1]^B$, and the goal is to estimate $\boldsymbol{v} = \sum_{i=1}^{n} \boldsymbol{v_i}$. Note that this problem degenerates into real summation by setting $B = 1$, or frequency estimation by setting $w_i = 1$ for all i. However, unlike the frequency estimation problem where we are concerned with the maximum error over all elements (the coordinates of $\boldsymbol{v}$ in this case), for the vector summation problem, the error metric is often the MSE, i.e., $\mathbb{E}[(\boldsymbol{v} - \hat{\boldsymbol{v}})^2]$ where $\hat{\boldsymbol{v}}$ is an estimated $\boldsymbol{v}$.

By directly applying their $1 + o(1)$ real summation protocol on each coordinate, Ghazi et al. [27] show how to achieve an MSE of $\tilde{O}(\min\{B, n\})$, matching the optimal central-DP error bound [11] with $1 + \tilde{O}(\sqrt{B/n} + B/n)$ messages per user[2]. This is a $1 + o(1)$-message protocol only in the low-dimensional setting $B = o(n)$. In this paper, we show that in high dimensions $B = \Omega(n)$, this problem can be solved by simply combining our frequency estimation and heavy hitter detection protocols, which can achieve $\tilde{O}(n)$ MSE with $1 + o(1)$ messages. Thus, together with [27], we have shown that the optimal central-DP MSE (up to polylogarithmic factors) can be achieved with $1 + o(1)$ messages in shuffle-DP.

2 RELATED WORK

The idea of anonymizing/shuffling the messages before handing them to an analyzer goes back to at least 1981 [13]. Combining this notion with differential privacy results in the shuffle-DP model, which has attracted much attention in recent years. In addition to the problems already covered earlier, Chen et al. [14] study the distinct count problem in shuffle-DP, which also exhibits a separation between the single-message and multi-message setting. They prove an error lower bound of $\Omega(n)$ (hence, hopeless) for the former, and an upper bound of $\tilde{O}(\sqrt{n})$ for the latter. The standard shuffle-DP model, like local-DP, only allows a single round of communication. Recently, Beimel et al. [8] extend this model and investigate what can be achieved with two rounds. Huang et al. [29] design a 3-round mean estimation protocol that achieves instance optimality. In this paper, we use the standard 1-round model of shuffle-DP.

The protocol in [25] is the state of the art for frequency estimation and heavy hitter detection for a large B, i.e., the dependency on B is logarithmic. Prior to that, there are a number of protocols with a linear dependency on B. Balcer and Cheu [2] provide a protocol that sends $O(B)$ messages per user while achieving $O(\log n)$ error, which keeps independent of the domain size even when $\log B = \Omega(\log n)$. Cheu and Zhilyaev [16] design a protocol that can send 2 messages per user, but each message has $O(B)$ bits. Ghazi et al. [26] show how to achieve $O(\log n)$ error with $1 + \tilde{O}(B/n)$ messages each of $O(\log B)$ bits. In Section 3.2 we show how our framework also can recover this result easily. Frequency estimation and heavy hitter detection have also been studied extensively in local-DP [1, 6, 7, 22, 23, 39, 41], but the optimal error in local-DP is $\tilde{\Theta}(\sqrt{n})$.

Compared with local-DP, shuffle-DP allows us to achieve a much smaller error, but it relies on the availability of an anonymizer or shuffler. From its early conception [13], many practical implementations of shufflers based on different technologies have been released, including mix networks [13, 19], onion routing [20, 35], trusted nodes/hardware [10, 36], etc.

3 FREQUENCY ESTIMATION

3.1 A Balls-into-bins mechanism

We first study the privacy of a simple balls-into-bins mechanism $\mathcal{M}^{\text{BIB}}$, shown in Algorithm 1, in the central-DP model. This mechanism, as we will see shortly, is equivalent to our shuffle-DP mechanisms with appropriate settings of its parameters m, s, k, n, p. The input to $\mathcal{M}^{\text{BIB}}$ is some $S \subseteq [m]$ such that $|S| = s$, and any pair of inputs S and S' are considered neighbors. The set S can be thought of as s special bins, chosen out of a total of m bins. The mechanism throws one *real* ball and (expected) $k + np$ *noisy* balls into the m bins, and the numbers of balls in all the bins are taken as the output. Specifically, it throws the real ball into one of the s special bins uniformly at random. Then, it throws k noisy balls uniformly at random into all the bins. Finally, it flips a biased coin n times (we use $\text{Ber}(p)$ to denote a Bernoulli random variable with parameter p in the algorithm), and for each heads we get, it throws a noisy ball into one bin chosen uniformly at random from all bins. All the balls are thrown independently. Clearly, for any $S \neq S'$, the distributions of the noisy balls are identical, whose purpose is to hide the location of the real ball. Since the location of the real ball is randomly chosen from the s special bins, a larger s makes it easier to hide (in the extreme case $s = m$, no noisy ball is needed). The theorem below formalizes this intuition:

Theorem 3.1. *For any $0 < \varepsilon \leq 3$ and $0 < \delta < 1$, the mechanism $\mathcal{M}^{\text{BIB}}$ is (ε, δ)-differentially private if $k + np \geq \frac{32 \ln(2/\delta)}{\varepsilon^2} \cdot \frac{m}{s}$.*

Proof. The proof follows the framework of Lemma 4.4 in [25]. Treating the numbers of balls in the m bins as an m-dimensional vector, we use $\mathcal{M}(S) \in \mathbb{N}^m$ to denote the output of $\mathcal{M}^{\text{BIB}}$ with public parameters m, s, k, n, p on input S. We prove the following inequality, which is equivalent to the DP definition in (1):

$$\Pr_{\boldsymbol{W} \sim \mathcal{M}(S)} \left[\frac{\Pr[\mathcal{M}(S) = \boldsymbol{W}]}{\Pr[\mathcal{M}(S') = \boldsymbol{W}]} \geq e^{\varepsilon} \right] \leq \delta. \tag{2}$$

We first consider mechanism $\mathcal{M}_0^{\text{BIB}}$ that only throws noisy balls (i.e., skip the first two lines in Algorithm 1). Let $\mathcal{M}_0 \in \mathbb{N}^m$ be the output vector of $\mathcal{M}_0^{\text{BIB}}$; note that $\mathcal{M}_0$ does not depend on S. For any $\boldsymbol{W} = (w_i)_{i=1}^m \in \mathbb{N}^m$, denote $w = \sum_{i=1}^m w_i$. Then when $k \leq w \leq k+n$, we have

$$\Pr[\mathcal{M}_0 = \boldsymbol{W}] = \binom{n}{w-k} p^{w-k} (1-p)^{n-w+k} \frac{w!}{\prod_{i=1}^m w_i!}.$$

Now consider $\mathcal{M}^{\text{BIB}}$. Let I be the bin chosen by the real ball (line 1 in Algorithm 1) and $\boldsymbol{e_i}$ be a length-m one-hot vector where the i-th coordinator is 1 and other coordinators are all 0. For any

[2] As stated, their protocol only achieves MSE $\tilde{O}(B)$ and the message number is $1 + \tilde{O}(B/\sqrt{n})$. In Section 5.1, we show how their protocol can be modified to achieve these improved bounds.

Algorithm 1: Balls-into-bins Mechanism $\mathcal{M}^{\text{BIB}}$

Public Parameters: m, s, k, n, p
Input: A set $S \subseteq [m]$ such that $|S| = s$
Output: A multiset $O \subseteq [m]$

Choose $x \in S$ uniformly at random;
$O \leftarrow \{x\}$;
for $i \leftarrow 1$ **to** k **do**
  Choose $x \in [m]$ uniformly at random;
  $O \leftarrow O \uplus \{x\}$; // Increase the multiplicity of x by 1 ($\uplus$ stands for the union operation on multisets).
end
for $i \leftarrow 1$ **to** n **do**
  $y \leftarrow \text{Ber}(p)$;
  if $y = 1$ **then**
    Choose $x \in [m]$ uniformly at random;
    $O \leftarrow O \uplus \{x\}$;
  end
end
return O;

$\boldsymbol{W} = (w_i)_{i=1}^m \in \mathbb{N}^m$, denote $w = \sum_{i=1}^m w_i$. Then when $1 + k \le w \le 1 + k + n$, we have

$$\begin{aligned}
&\Pr[\mathcal{M}(S) = \boldsymbol{W}] \\
=&\sum_{i \in S} \Pr[\mathcal{M}(S) = \boldsymbol{W} \mid I = i] \cdot \Pr[I = i] \\
=&\sum_{i \in S} \Pr[\mathcal{M}_0 = \boldsymbol{W} - \boldsymbol{e}_i] \cdot s^{-1} \\
=&\binom{n}{w-1-k} p^{w-1-k}(1-p)^{n-w+1+k} \cdot \frac{(w-1)!}{s\prod_{i=1}^m w_i!} \sum_{i \in S} w_i.
\end{aligned}$$

Therefore, inequality (2) is equivalent to

$$\Pr_{\boldsymbol{W} \sim \mathcal{M}(S)}\left[\frac{\sum_{i \in S} w_i}{\sum_{i \in S'} w_i} \ge e^{\varepsilon}\right] \le \delta.$$

We use $\text{Bin}(n, p)$ to denote a binomial distribution with parameters n and p. Notice that when $\boldsymbol{W} \sim \mathcal{M}(S)$, $\sum_{i \in S} w_i \sim 1 + \text{Bin}(k, \frac{s}{m}) + \text{Bin}(n, \frac{ps}{m})$, while $\sum_{i \in S'} w_i \sim \text{Ber}(|S \cap S'|/s) + \text{Bin}(k, \frac{s}{m}) + \text{Bin}(n, \frac{ps}{m})$. Therefore, it suffices to prove

$$\Pr\left[\frac{1 + X_1}{X_2} \ge e^{\varepsilon}\right] \le \delta, \tag{3}$$

where $X_1, X_2 \sim \text{Bin}(k, \frac{s}{m}) + \text{Bin}(n, \frac{ps}{m})$. Note that X_1 and X_2 are not necessarily independent.

Since $\mu := \mathbb{E}[X_1] = (k + np)\frac{s}{m} \ge \frac{32\ln(2/\delta)}{\varepsilon^2}$, by Chernoff bound, we have

$$\Pr[X_1 \ge (1 + \varepsilon/3)\mu] \le \exp\left(-\frac{\mu}{3}\left(\frac{\varepsilon}{3}\right)^2\right) < \frac{\delta}{2},$$

$$\Pr[X_2 \le (1 - \varepsilon/4)\mu] \le \exp\left(-\frac{\mu}{2}\left(\frac{\varepsilon}{4}\right)^2\right) < \frac{\delta}{2}.$$

Since $\delta < 1$, we have $\mu > (32\ln 2)/\varepsilon^2 > 18/\varepsilon^2$. Also note that $e^{\varepsilon/3} \ge 1 + \varepsilon/3 + \varepsilon^2/18$ and $1 - \varepsilon/4 > e^{-\varepsilon/2}$ for $0 < \varepsilon \le 3$, so we have

$$\frac{1 + (1 + \varepsilon/3)\mu}{(1 - \varepsilon/4)\mu} \le \frac{\varepsilon^2/18 + (1 + \varepsilon/3)}{1 - \varepsilon/4} \le \frac{e^{\varepsilon/3}}{e^{-\varepsilon/2}} \le e^{\varepsilon}.$$

Finally, by a union bound, we obtain

$$\Pr\left[\frac{1 + X_1}{X_2} \ge e^{\varepsilon}\right] \le \Pr[X_1 \ge (1+\varepsilon/3)\mu] + \Pr[X_2 \le (1-\varepsilon/4)\mu] \le \delta.$$

□

Remark. From the proof of Theorem 3.1, we see that the derivation from

$$(k + np) \cdot \frac{s}{m} \ge \theta := \frac{32\ln(2/\delta)}{\varepsilon^2} \tag{4}$$

to the key inequality (3) is quite loose. In particular, we chose a large constant 32 so as to simplify the proof. In practice, when concrete values of $n, m, s, \varepsilon, \delta$ are given, we can try minimizing θ as long as (3) holds. Specifically, we consider the worst case $S \cap S' = \emptyset$, in which case $\Pr[(1 + X_1)/X_2 \ge e^{\varepsilon}]$ is a function of θ that can be computed in $O(n)$ time. Then, we perform a binary search for the smallest θ such that (3) holds. We stop the binary search when the range has narrowed down to $1/n$, which gives us a near-optimal θ in $O(n \log n)$ time. Note that this optimization step is data-independent, so θ can be pre-computed. Furthermore, it works without the technical condition $0 < \varepsilon \le 3$ in Theorem 3.1.

3.2 The Protocol for a Small Domain

As a warm-up, we first describe a protocol for a small domain $B < \tilde{O}(n)$. It is a simple modification of the single-message shuffle protocol of [4]. In their local randomizer, each user sends its input with probability $\rho = \tilde{O}(B/n)$, and sends a *blanket noise* otherwise. The blanket noise is a random element uniformly chosen from $[B]$. When $B = \sqrt{n}$, their analyzer incurs $\tilde{O}(n^{1/4})$ error, matching the lower bound of single-message protocols [25].

We extend their result to a $(1 + \rho)$-message protocol $\mathcal{P}^{\text{FE0}} = (\mathcal{R}^{\text{FE0}}, \mathcal{A}^{\text{FE0}})$ for better accuracy, where $\rho = \tilde{O}(B/n)$. The local randomizer works as follows: Each user always sends its input; in addition, with probability ρ, it also sends a blanket noise (see Algorithm 2).

Algorithm 2: Local Randomizer $\mathcal{R}^{\text{FE0}}$

Public Parameters: $B, n, \varepsilon, \delta$
Input: $x \in [B]$
Output: A multiset $\mathcal{T} \subseteq [B]$

$\mathcal{T} \leftarrow \{x\}$;
$\rho \leftarrow \frac{32\ln(2/\delta)}{\varepsilon^2} \cdot \frac{B}{n}$;
$y \leftarrow \text{Ber}(\rho)$;
if $y = 1$ **then**
  Choose x' from $[B]$ uniformly;
  $\mathcal{T} \leftarrow \mathcal{T} \uplus \{x'\}$;
end
return $\mathcal{T}$;

We first show that this local randomizer satisfies DP:

Lemma 3.2 (Privacy of $\mathcal{R}^{\mathrm{FE0}}$). *For any $n, B \in \mathbb{N}$, $0 < \varepsilon \le 3$, and $0 < \delta < 1$, let $\rho = \frac{32\ln(2/\delta)}{\varepsilon^2} \cdot \frac{B}{n}$. If $\rho \le 1$, then $\mathcal{R}^{\mathrm{FE0}}$ satisfies (ε, δ)-shuffle DP.*

Proof. For any input $\boldsymbol{D} = (x_1, x_2, \ldots, x_n)$, the view of the analyzer is the multiset $\mathcal{T}$ which can be split into two parts:

(1) n real inputs from all the users $\{x_i\}_{i=1}^n$, and
(2) the multiset of expected $n\rho$ blanket noises $\mathcal{T}_{\mathrm{Noise}} \subseteq [B]$.

Let $\boldsymbol{D'}$ be any neighbouring databases of $\boldsymbol{D}$. Then we need to prove:

$$\Pr_{\boldsymbol{\tau}\sim\mathcal{T}(\boldsymbol{D})}\left[\frac{\Pr[\mathcal{T}(\boldsymbol{D})=\boldsymbol{\tau}]}{\Pr[\mathcal{T}(\boldsymbol{D'})=\boldsymbol{\tau}]} \ge e^{\varepsilon}\right] \le \delta. \tag{5}$$

Without loss of generality, we assume $\boldsymbol{D}$ and $\boldsymbol{D'}$ differ in the last element, i.e., $\boldsymbol{D'} = (x_1, x_2, \ldots, x_n')$. By factoring out the first $n-1$ users, it suffices to prove:

$$\Pr_{\boldsymbol{\tau}\sim\mathcal{T}_{\mathrm{Noise}}}\left[\frac{\Pr[\mathcal{T}_{\mathrm{Noise}}=\boldsymbol{\tau}]}{\Pr[\mathcal{T}_{\mathrm{Noise}}\uplus\{x_n'\}=\boldsymbol{\tau}\uplus\{x_n\}]} \ge e^{\varepsilon}\right] \le \delta. \tag{6}$$

Note that (6) is actually stronger than (5), as the first $n-1$ common users provide additional randomness. To prove (6), we note that without real inputs from the first $n-1$ users, the shuffled messages correspond to $\mathcal{M}^{\mathrm{BIB}}$ with public parameters $m = B, s = 1, k = 0, n, p = \rho$:

- The set of bins in $\mathcal{M}^{\mathrm{BIB}}$ corresponds to $[B]$.
- The input set S in $\mathcal{M}^{\mathrm{BIB}}$ corresponds to $\{x_n\}$ and $\{x_n'\}$, respectively.
- No fixed k noisy balls, i.e., $k = 0$.
- The multiset of bins chosen by the expected np noisy balls are $\mathcal{T}_{\mathrm{Noise}}$.

Then the lemma immediately follows from Theorem 3.1. □

For a query x, the analyzer (Algorithm 3) simply counts the number of messages that equal to x, followed by a bias-removal step.

Algorithm 3: Analyzer $\mathcal{A}^{\mathrm{FE0}}$

Public Parameters: $B, n, \varepsilon, \delta$
Input: A multiset $\mathcal{T} \subseteq [B]$; element x
Output: Estimated frequency of x

$X \leftarrow$ the frequency of x in $\mathcal{T}$;
$\rho \leftarrow \frac{32\ln(2/\delta)}{\varepsilon^2} \cdot \frac{B}{n}$;
$\hat{g}_x \leftarrow X - n\rho/B$;
return $\hat{g}_x$;

Lemma 3.3 (Accuracy of $\mathcal{P}^{\mathrm{FE0}}$). *The protocol $\mathcal{P}^{\mathrm{FE0}}$ has error*

$$\alpha = \max\left\{3\ln(2B/\beta), \sqrt{3\ln(2B/\beta)\cdot\frac{32\ln(2/\delta)}{\varepsilon^2}}\right\}.$$

Proof. Let X be the frequency of x in the output multiset $\mathcal{T}$, then $X = g_x + \mathrm{Bin}(n, \rho/B)$, where the second term comes from the blanket noises. Since $\hat{g}_x = X - n\rho/B$, we conclude $\mathbb{E}[\hat{g}_x] = g_x$, so $\mathcal{A}^{\mathrm{FE0}}$ provides an unbiased estimation.

Let $Y = \mathrm{Bin}(n, \rho/B)$. Define $\mu \coloneqq \mathbb{E}[Y]$, then by Chernoff bound, for $0 < \eta \le 1$,

$$\Pr\left[|Y-\mu| > \eta\mu\right] < 2\exp(-\frac{\eta^2\mu}{3}).$$

And for $\eta > 1$,

$$\Pr\left[|Y-\mu| > \eta\mu\right] < 2\exp(-\frac{\eta\mu}{3}).$$

If $\mu \ge 3\ln(2B/\beta)$, by setting $\eta = \sqrt{\frac{3\ln(2B/\beta)}{\mu}} \le 1$ we get

$$\Pr\left[|Y-\mu| > \sqrt{3\ln(2B/\beta)\mu}\right] < 2\exp(-\frac{\eta^2\mu}{3}) = \frac{\beta}{B}.$$

Otherwise, by setting $\eta = \frac{3\ln(2B/\beta)}{\mu} > 1$ we get

$$\Pr\left[|Y-\mu| > 3\ln(2B/\beta)\right) < 2\exp(-\frac{\eta\mu}{3}) = \frac{\beta}{B}.$$

In conclusion,

$$\Pr\left[|Y-\mu| > \max\left\{3\ln(2B/\beta), \sqrt{3\ln(2B/\beta)\cdot\mu}\right\}\right] < \frac{\beta}{B}.$$

Therefore,

$$\Pr\left[|\hat{g}_x - g_x| > \alpha\right] < \frac{\beta}{B}.$$

By union bound,

$$\Pr\left[\bigvee_{x\in[B]} |\hat{g}_x - g_x| > \alpha\right] < \beta,$$

which implies the result. □

The number of messages sent per user, message size, and query time are obvious. Thus we have:

Theorem 3.4. *For $0 < \varepsilon \le 3$, $0 < \delta < 1$, and any $n, B \in \mathbb{N}$ such that $B \le \frac{\varepsilon^2 n}{32\ln(2/\delta)}$, $\mathcal{P}^{\mathrm{FE0}}$ is a private-coin (ε, δ)-shuffle DP frequency estimation protocol that sends $1 + O\left(\frac{\log(1/\delta)}{\varepsilon^2}\cdot\frac{B}{n}\right)$ messages per user in expectation, each consisting of $O(\log B)$ bits. Any frequency query can be answered in expected $O(n)$ time with error $O\left(\log\frac{B}{\beta} + \frac{1}{\varepsilon}\sqrt{\log\frac{B}{\beta}\log(1/\delta)}\right)$.*

Note that for $B = o\left(\frac{\varepsilon^2 n}{\log(1/\delta)}\right)$, $\mathcal{P}^{\mathrm{FE0}}$ is a $(1 + o(1))$-message protocol with error $\tilde{O}(1)$.

3.3 The Protocol for a Large Domain

The protocol above can be easily extended to support a larger domain. For $B \ge \tilde{\Omega}(n)$, ρ will be greater than 1, but we can ask each user to first send $\lfloor\rho\rfloor$ blanket noises, and then another blanket noise with probability $\rho - \lfloor\rho\rfloor$. Privacy still holds, by appropriately modifying the proof of Lemma 3.2, but the number of messages will be $1 + \rho = \tilde{O}(B/n)$. In this section, we provide another protocol $\mathcal{P}^{\mathrm{FE1}}$, which sends $1 + o(1)$ messages for an arbitrarily large B while achieving $\tilde{O}(1)$ error.

The idea is to use hashing to reduce the domain size. Specifically, we use the classical hash function $h_{u,v} : [B] \to [b]$ to hash an element in $[B]$ to a bin in $[b]$, where $h_{u,v}(x) = ((ux + v) \bmod q) \bmod b$, $q \in [B, 2B)$ is a prime[3], and b is a parameter that will

[3] This q must exist due to Bertrand's postulate.

control the trade-off between communication cost and accuracy. Let $\mathcal{H} = \{h_{u,v} \mid (u,v) \in \{1,\dots,q-1\} \times [q]\}$. It is well-known that $\mathcal{H}$ is a universal family with collision probability $p_{\text{col}}(x,y) \coloneqq \Pr[h_{u,v}(x) = h_{u,v}(y)] < \frac{1}{b}$. In fact, for this family, the collision probability is the same for all $x \neq y$: $p_{\text{col}}(\cdot,\cdot) \equiv \lfloor q/b \rfloor((q \bmod b) + q - b)/(q(q-1))$, which is simply denoted as p_{col}.

3.3.1 Local randomizer (Algorithm 4). Assume a user holds a private input x. It uniformly randomly chooses $(u,v) \in \{1,\dots,q-1\} \times [q]$, and sends $(u,v,h_{u,v}(x))$ to the shuffler, called the *real output*. It also sends ρ *blanket noises* to the shuffler, where $\rho = \frac{32\ln(2/\delta)}{\varepsilon^2} \cdot \frac{b}{n}$. Each blanket noise (u,v,w) is uniformly chosen from $\{1,\dots,q-1\} \times [q] \times [b]$ independently. Note that when ρ is not an integer, it sends $\lfloor \rho \rfloor$ blanket noises and then with probability $\rho - \lfloor \rho \rfloor$ sends another blanket noise, so that the expected number of blanket noises is ρ. The total expected number of messages the user sends is therefore $1 + \rho$, including one real output and ρ blanket noises.

Algorithm 4: Local Randomizer $\mathcal{R}^{\text{FE1}}$

Public Parameters: $B, b, n, \varepsilon, \delta$
Input: $x \in [B]$
Output: A multiset $\mathcal{T} \subseteq \{1,\dots,q-1\} \times [q] \times [b]$

Choose (u,v) from $\{1,\dots,q-1\} \times [q]$ uniformly;
$\mathcal{T} \leftarrow \{(u,v,h_{u,v}(x))\}$;
$\rho \leftarrow \frac{32\ln(2/\delta)}{\varepsilon^2} \cdot \frac{b}{n}$;
for $i \leftarrow 1$ **to** $\lfloor \rho \rfloor$ **do**
  Choose (u,v,w) from $\{1,\dots,q-1\} \times [q] \times [b]$ uniformly;
  $\mathcal{T} \leftarrow \mathcal{T} \uplus \{(u,v,w)\}$;
end
$y \leftarrow \text{Ber}(\rho - \lfloor \rho \rfloor)$;
if $y = 1$ **then**
  Choose (u,v,w) from $\{1,\dots,q-1\} \times [q] \times [b]$ uniformly;
  $\mathcal{T} \leftarrow \mathcal{T} \uplus \{(u,v,w)\}$;
end
return $\mathcal{T}$;

Lemma 3.5 (Privacy of $\mathcal{R}^{\text{FE1}}$). *For any $n, B, b \in \mathbb{N}$, $0 < \varepsilon \le 3$, and $0 < \delta < 1$, let $\rho = \frac{32\ln(2/\delta)}{\varepsilon^2} \cdot \frac{b}{n}$. Then $\mathcal{R}^{\text{FE1}}$ satisfies (ε,δ)-shuffle DP.*

Proof. The proof is similar to the proof of Lemma 3.2: First we exclude $\{(u_i, v_i, h_{u_i,v_i}(x_i))\}_{i=1}^{n-1}$, i.e., the real outputs from the first $n-1$ users. Then the privacy follows from that of $\mathcal{M}^{\text{BIB}}$ with parameters $m = (q-1)qb, s = (q-1)q, k = n\lfloor \rho \rfloor, n, p = \rho - \lfloor \rho \rfloor$:

- The set of bins in $\mathcal{M}^{\text{BIB}}$ are $\{1,\dots,q-1\} \times [q] \times [b]$.
- The input set S in $\mathcal{M}^{\text{BIB}}$ corresponds to $\{(u,v,w) \mid h_{u,v}(w) = x_n\}$ and $\{(u,v,w) \mid h_{u,v}(w) = x_n'\}$, respectively.
- Each user first throws $\lfloor \rho \rfloor$ noisy balls, so $k = n\lfloor \rho \rfloor$.
- Each user then throws a noisy ball with probability $p = \rho - \lfloor \rho \rfloor$. □

3.3.2 Analyzer (Algorithm 5). The analyzer's view is a multiset of tuples (u,v,w). To query for the frequency of an element x, it scans all tuples received and count the number of tuples satisfying $h_{u,v}(x) = w$. Then we remove the bias caused by hash collisions and the blanket noises.

Algorithm 5: Analyzer $\mathcal{A}^{\text{FE1}}$

Public Parameters: $B, b, n, \varepsilon, \delta$
Input: An element $x \in [B]$; a multiset $\mathcal{T} \subseteq \{1,\dots,q-1\} \times [q] \times [b]$
Output: Estimated frequency of x

$p_{\text{col}} \leftarrow \lfloor q/b \rfloor((q \bmod b) + q - b)/(q(q-1))$;
$X \leftarrow 0$;
for $(u,v,w) \in \mathcal{T}$ **do**
  if $h_{u,v}(x) = w$ **then**
    $X \leftarrow X + 1$
  end
end
$\rho \leftarrow \frac{32\ln(2/\delta)}{\varepsilon^2} \cdot \frac{b}{n}$;
$\hat{g}_x \leftarrow (X - n\rho/b - np_{\text{col}})/(1 - p_{\text{col}})$;
return $\hat{g}_x$;

Lemma 3.6 (Accuracy of $\mathcal{P}^{\text{FE1}}$). *The protocol $\mathcal{P}^{\text{FE1}}$ has error*

$$\alpha = 2\max\left\{3\ln(2B/\beta), \sqrt{3\ln(2B/\beta) \cdot \left(\frac{n}{b} + \frac{32\ln(2/\delta)}{\varepsilon^2}\right)}\right\}.$$

Proof. Let X be the number of tuples (u,v,w) in $\mathcal{T}$ that satisfy $h_{u,v}(x) = w$. We have $X = g_x + \text{Bin}(n - g_x, p_{\text{col}}) + \text{Bin}(n\lfloor \rho \rfloor, 1/b) + \text{Bin}(n, (\rho - \lfloor \rho \rfloor)/b)$, where

(1) the first term is the g_x real outputs from the users with input x,
(2) the second term is from the $n - g_x$ real outputs from the users with data not equal to x, each colliding with x with probability p_{col}, and
(3) the last two terms are from the blanket noises.

Since $\hat{g}_x = (X - n\rho/b - np_{\text{col}})/(1 - p_{\text{col}})$, we have $\mathbb{E}[\hat{g}_x] = g_x$, i.e., $\mathcal{P}^{\text{FE1}}$ returns unbiased estimates.

Let $Y = \text{Bin}(n - g_x, p_{\text{col}}) + \text{Bin}(n\lfloor \rho \rfloor, 1/b) + \text{Bin}(n, (\rho - \lfloor \rho \rfloor)/b)$ which is a sum of $n(\lfloor \rho \rfloor + 2) - g_x$ independent Bernoulli variables. Define $\mu \coloneqq \mathbb{E}[Y]$, then similar to the proof of Lemma 3.3, we have

$$\Pr\left[|Y - \mu| > \max\left\{3\ln(2B/\beta), \sqrt{3\ln(2B/\beta) \cdot \mu}\right\}\right] < \frac{\beta}{B}.$$

Therefore,

$$\Pr\left[|\hat{g}_x - g_x| > \max\{3\ln(2B/\beta), \sqrt{3\ln(2B/\beta) \cdot \mu}\} / (1 - p_{\text{col}})\right] < \frac{\beta}{B}. \tag{7}$$

Recall that $p_{\text{col}} < 1/b$, we have

$$\mu = p_{\text{col}}(n - g_x) + n\rho/b \le \frac{n}{b} + \frac{32\ln(2/\delta)}{\varepsilon^2}.$$

Also note that $p_{\text{col}} \le 1/2$, so inequality (7) can be relaxed to

$$\Pr[|\hat{g}_x - g_x| > \alpha] < \frac{\beta}{B}.$$

Protocol		Messages per user	Message size	Error	Query time (single)	Query time (all)
CZ [16]		2	B	$O(\log n)$	$O(n)$	$O(Bn)$
		2	$O(\log n + B\log^2 n/n)$	$O(\log n)$	$O(n\log n)$	$O(n\log n + B\log^2 n)$
		$O(\log n)$	$O(\log^5 n)$	$O(\log^2 n)$	$O(n\log n)$	$O(n\log n + B\log n)$
GGKPV [25]		$O(\log n)$	$O(\log^2 n)$	$O(\log n)$	$O(n\log^3 n)$	$O(n\log^4 n + B\log^2 n)$
This work (small B)		$1 + O(B\log n/n)$	$O(\log n)$	$O(\log n)$	$O(n)$	$O(n)$
This work (large B)		$1 + O(b\log n/n)$	$O(\log n)$	$O(\log n + \sqrt{\frac{n}{b}\log n})$	$O(n + b\log n)$	$O(Bn/b + B\log n)$
Above with	$c = 1$	$O(1)$	$O(\log n)$	$O(\log n)$	$O(n)$	$O(B\log n)$
$b = \frac{n}{\log^c n}$	$c > 1$	$1 + o(1)$	$O(\log n)$	$O(\log^{\frac{c+1}{2}} n)$	$O(n)$	$O(B\log^c n)$
Corollary 4.5 ($c > 2$)		$1 + o(1)$	$O(\log n)$	$O(\log^c n)$	-	$O(n^2/\log^c n)$

Table 1: Comparison with prior works for frequency estimation, assuming $\varepsilon = \Theta(1)$ and $\log n = \Theta(\log(1/\delta)) = \Theta(\log B) = \Theta(\log(1/\beta))$.

Then the lemma follows by a union bound. □

Theorem 3.7. *For any $0 < \varepsilon \le 3$, $0 < \delta < 1$, and $n, B, b \in \mathbb{N}$ such that $2 \le b \le B/2$, $\mathcal{P}^{\mathrm{FE1}}$ is a private-coin (ε,δ)-shuffle DP frequency estimation protocol that sends $1+O\left(\frac{\log(1/\delta)}{\varepsilon^2}\cdot\frac{b}{n}\right)$ messages per user in expectation, each consisting of $O(\log B)$ bits. The frequency query of any element $x \in [B]$ can be answered in expected $O\left(n + \frac{\log(1/\delta)}{\varepsilon^2}\cdot b\right)$ time with error $O\left(\log\frac{B}{\beta} + \sqrt{\log\frac{B}{\beta}\left(\frac{n}{b} + \frac{\log(1/\delta)}{\varepsilon^2}\right)}\right)$. The frequencies of all elements in $[B]$ can be found in expected $O\left(\left(\frac{n}{b} + \frac{\log(1/\delta)}{\varepsilon^2}\right)\cdot B\right)$ time.*

Proof. It only remains to prove the last statement, which is faster than querying each element one by one by a factor of b. Such a faster all-frequency algorithm will also be useful for heavy hitter detection, which in turn will be needed for sparse vector summation.

The idea is that, instead of computing the estimated frequency of each $x \in [B]$, we flip the problem around. Recall that the variable X in Algorithm 5 is equal to the number of (u,v,w) tuples in $\mathcal{T}$ such that $h_{u,v}(x) = w$. We initialize this variable to 0 for all $x \in [B]$. Then, for each tuple $(u,v,w) \in \mathcal{T}$, we find the set of all $x \in [B]$ such that $h_{u,v}(x) = w$ and increment their corresponding variables. It can be verified that this set is

$$\mathcal{X}(u,v,w) := \left\{u^{-1}(w + i\cdot b - v) \bmod q \;\middle|\; i \in \left\{0,1,\ldots,\left\lfloor\frac{q-1-w}{b}\right\rfloor\right\}\right\} \cap [B],$$

where u^{-1} is the multiplicative inverse of u in $\mathbb{Z}_q$. The set $\mathcal{X}(u,v,w)$ has at most $q/b = O(B/b)$ elements and they can be found in $O(B/b)$ time. Meanwhile $\mathbb{E}[|\mathcal{T}|] = n + n\rho = O\left(n + b\cdot\frac{\ln(1/\delta)}{\varepsilon^2}\right)$, so the total running time is $O\left(\left(\frac{n}{b} + \frac{\log(1/\delta)}{\varepsilon^2}\right)\cdot B\right)$. □

We compare our protocols with the state-of-the-art frequency estimation protocol [25] in Table 1. To make the bounds more concise, we set a constant ε, and consider the parameter range $\Theta(\log(1/\delta)) = \Theta(\log n) = \Theta(\log B) = \Theta(\log(1/\beta))$. In particular, by taking $b = n/\log n$ in $\mathcal{P}^{\mathrm{FE1}}$, our protocol achieves the same error as [25], but we reduce the number of messages to $O(1)$, reduce the message size by a $\log n$ factor, and reduce the query time of a single element by a $\log^3 n$ factor. Recall that we require $b < B/2$, so for $B < 2n/\log n$, we use the small-domain protocol, which obtains the same (or better) improvement. By using a smaller b, the number of messages is further reduced to $1 + o(1)$, although at the expense of a small increase in the error. Furthermore, by imposing a dyadic decomposition on the domain, a frequency estimation protocol can be used for solving the *range-counting* problem, so our result immediately improves the range-counting results in [25] as well. We omit the rather tedious details.

3.3.3 Robust shuffle-DP. Since shuffle-DP only uses one-way communications, it naturally defends against a semi-honest adversary that may compromise any number of users. In the robust $(\varepsilon,\delta,\gamma)$ shuffle-DP model [3], a fraction of $1-\gamma$ of the users are allowed to be malicious, i.e., they may not follow the specified protocol. Our protocol can be easily made robust, since each users injects (expected) ρ blanket noises independently. By setting $\rho = \frac{32\ln(2/\delta)}{\gamma\varepsilon^2}\cdot\frac{b}{n}$, we can still ensure that sufficient blanket noises are added. Nevertheless, it is not possible to guarantee accuracy if users deviate from the protocol, as the malicious users may replace their data by fake values. The same situation happens for the MPC model with malicious parties, which can only guarantee privacy but not utility.

4 HEAVY HITTER DETECTION

A standard technique to reduce the heavy hitter detection problem to frequency estimation is to use a prefix tree [17, 21] of $\log B$ levels over the domain $[B]$. For any $x \in [B]$, we use $x[:i]$ to denote the first i bits in the binary representation of x. For level i of the prefix tree, we map every $x \in [B]$ to $x[:i] \in [2^i]$, and construct a frequency estimation data structure over the domain $[2^i]$. It is clear that $x[:i]$ is a heavy hitter in level i only if $x[:(i-1)]$ is a

heavy hitter in level $i-1$. To identify all the heavy hitters, we start from level 1, which has only 2 elements in its domain, query for their frequencies, and then proceed to lower levels, only querying for the frequencies of elements whose prefixes are heavy hitters. Adopting this idea directly for shuffle-DP, as done in [25], leads to an $O(\log^3 B)$-factor increase in the message number, because the privacy budget ε has to be divided among the $\log B$ levels.

We improve their heavy hitter detection algorithm using the following two ideas. First, instead of participating in all levels of the prefix tree, we ask each user to randomly pick one level. This reduces the message number to $O(1)$. A similar idea was used in the local-DP protocol [6], but our shuffle-DP protocol only uses private randomness where [6] needs public randomness. Next, we sample the messages to further reduce the message number to $o(1)$.

Below we present our heavy hitter detection protocol $\mathcal{P}^{\text{HHD}} = (\mathcal{R}^{\text{HHD}}, \mathcal{A}^{\text{HHD}})$ for the case $B > n/\log n$. If $B \le n/\log n$, we can simply use the small-domain frequency estimation protocol without the need of heavy hitter detection. We also assume B and $n/\log n$ are both powers of 2.

4.1 Local randomizer

Our heavy hitter detection randomizer $\mathcal{R}^{\text{HHD}}$ (Algorithm 6) works as follows. Let $s = \log(n/\log n)$ be the first level, $t = \log B$ be the last level, and $r = t - s + 1$ be the total number of levels. Suppose a user has element x. It first uniformly randomly chooses a level $i \in \{s, \dots, t\}$, and then inputs $x[:i]$ to the local randomizer of our frequency estimation protocol (with carefully chosen parameters, see Algorithm 6). Then it adds the label i to each message and sends it out with probability p, whose value will be determined by the analysis.

Algorithm 6: Local Randomizer $\mathcal{R}^{\text{HHD}}$

Public Parameters: $B, n, p, \varepsilon, \delta$
Input: $x \in [B]$
Output: A multiset $\mathcal{T} \subseteq \{s, \dots, t\} \times \{1, \dots, q-1\} \times [q] \times [b]$

$q \leftarrow$ the smallest prime that is greater or equal to B;
$b \leftarrow n/\log^2 n$;
$\mathcal{T} \leftarrow \emptyset$;
$i \leftarrow$ Uniformly chosen in $\{s, \dots, t\}$;
$\mathcal{R}_i \leftarrow \mathcal{R}^{\text{FE1}}$ with parameters $2^i, b, n/2r, \varepsilon, \delta/2$;
$\mathcal{T}_i \leftarrow \mathcal{R}_i(x[:i])$;
for $(u_j, v_x, w_j) \in \mathcal{T}_i$ **do**
  $y_j \leftarrow \text{Ber}(p)$;
  if $y_j = 1$ **then**
    $\mathcal{T} \leftarrow \mathcal{T} \uplus \{(i, u_j, v_x, w_j)\}$;
  end
end
return $\mathcal{T}$;

Lemma 4.1 (Privacy of $\mathcal{R}^{\text{HHD}}$). *$\mathcal{R}^{\text{HHD}}$ satisfies (ε, δ)-shuffle DP if $n \ge 8r \ln \frac{2r}{\delta}$.*

Proof. For any $i \in \{s, \dots, t\}$, let n_i be the number of users that choose this level, then $n_i \sim \text{Bin}(n, 1/r)$. By Chernoff bound, $\Pr[n_i < n/2r] \le \exp(-n/8r)$. By union bound, $\Pr[\vee_{i=s}^{t}(n_i < n/2r)] \le r\exp(-n/8r) \le \delta/2$, i.e., the number of users on every level is at least $n/2r$ except with probability at most $\delta/2$. Conditioned upon this happening, the messages at each level satisfies $(\varepsilon, \delta/2)$-shuffle DP by Theorem 3.1. The conditionality turns it into (ε, δ)-DP. Finally, since no user participates in more than one level, the privacy of all levels follows from parallel composition of DP. □

4.2 Analyzer

Now we introduce the analyzer $\mathcal{A}^{\text{HHD}}$ (Algorithm 8). First it classifies the messages to the correct levels according to their labels. Then starting from $C_s = \{0,1\}^s$, it recovers the heavy hitter candidates bit by bit. For each $i = s, \dots, t$, it counts the occurrences of all candidates using Algorithm 7, which is the same as the analyzer of our frequency estimation protocol but without bias removal. All candidates that appear less than Δ times are removed, and the remaining candidates are extended by one bit to obtain C_{i+1} except in the last level. The final set of candidates is denoted as H.

Algorithm 7: Counter C^{HHD}

Public Parameters: B, b
Input: An element x; A multiset $\mathcal{T} \subseteq \{1, \dots, q-1\} \times [q] \times [b]$
Output: Occurrences X of x

$q \leftarrow$ the smallest prime that is greater or equal to B;
$X \leftarrow 0$;
for $(u, v, w) \in \mathcal{T}$ **do**
  if $h_{u,v}(x) = w$ **then**
    $X \leftarrow X + 1$;
  end
end
return X;

Algorithm 8: Analyzer $\mathcal{A}^{\text{HHD}}$

Public Parameters: B, n, p
Input: A multiset $\mathcal{T} \subseteq \{s, \dots, t\} \times \{1, \dots, q-1\} \times [q] \times [b]$
Output: The set of heavy hitters candidates $H \subseteq [B]$

$q \leftarrow$ the smallest prime that is greater or equal to B;
$\Delta \leftarrow p\phi n/2r$;
for $i \leftarrow s$ **to** t **do**
  $\mathcal{T}_i \leftarrow \{(u, v, w) \mid (i, u, v, w) \in \mathcal{T}\}$;
end
$C_s \leftarrow \{0,1\}^s$;
for $i \leftarrow s$ **to** $t-1$ **do**
  Set the parameters of C^{HHD} to be $2^i, n/\log^2 n$;
  $C_i' \leftarrow \{x \in C_i \mid C^{\text{HHD}}(x, \mathcal{T}_i) \ge \Delta\}$;
  $C_{i+1} \leftarrow \{\overline{x0}, \overline{x1} \mid x \in C_i'\}$;
end
Set the parameters of C^{HHD} to be $2^t, n/\log^2 n$;
$H \leftarrow \{x \in C_t \mid C^{\text{HHD}}(x, \mathcal{T}_t) \ge \Delta\}$;
return H;

Lemma 4.2 (Completeness of $\mathcal{P}^{\mathrm{HHD}}$). *For any $\beta \in (0,1)$, if $p \geq \frac{8r}{\phi n} \ln \frac{r}{\phi\beta}$, then H contains all the heavy hitters with probability at least $1-\beta$.*

Proof. We use $g(x[:i])$ to denote the frequency of $x[:i]$ in the i-th level. Then for any heavy hitter x, $g(x[:i]) \geq \phi n$. Let X_i be the counter for x returned by C^{HHD}. We bound the probability $\Pr[X_i < \Delta]$. Since blanket noises and hash collisions can only make X_i larger, it suffices to bound $\Pr[\mathrm{Bin}(\phi n, p/r) < \Delta]$. By Chernoff bound, this is at most $\exp(-p\phi n/8r) \leq \phi\beta/r$. Then the probability that x is removed from the set of candidates follows from a union bound:

$$\Pr[x \notin H] = \Pr\left[\bigvee_{i=s}^{t} (X_i < \Delta)\right] \leq \sum_{i=s}^{t} \Pr[X_i < \Delta] \leq \phi\beta.$$

Finally, applying another union bound over all the at most $1/\phi$ heavy hitters yields the lemma. □

Lemma 4.3 (Efficiency of $\mathcal{P}^{\mathrm{HHD}}$). *Assume $\log n = \Theta(\log B) = \Theta(\log(1/\delta))$ and ε is a constant. If $\phi = \Omega(\log^2 n/n)$ and $p \geq \frac{30r}{\phi n} \ln(\phi B)$, then*

1. *$\mathbb{E}[|C_i|] = O(1/\phi)$ for each $i = s+1, \ldots, t$;*
2. *the expected number of messages per user is $O(p)$;*
3. *the expected running time of $\mathcal{A}^{\mathrm{HHD}}$ is $O(pn/\phi)$.*

Proof. First we prove (1). We bound Y, the number of elements whose occurrences exceed Δ. Let n_i be the number of users that have chosen level i. Define $n_0 := 2n/r$. By Chernoff bound, $\Pr[n_i > n_0] \leq \exp(-n/3r) = O(1/Bn)$. Next we assume $n_i \leq n_0$ for all i. Note that more users chosen in level i would increase Y, so we only need to consider the case $n_i = n_0$. We fix any combination of n_0 users and prove the result. For any $x \in \{0,1\}^i$, denote $g_i(x)$ as the true frequency of x in the n_0 users and X as the occurrences estimated by Algorithm 7. Then

$$\begin{aligned} X \sim\ & \mathrm{Bin}(g_i(x), p) + \mathrm{Bin}\left(n_0 - g_i(x), pp_{\mathrm{col}}\right) \\ & + \mathrm{Bin}\left(n_0\lfloor\rho\rfloor, p/b\right) + \mathrm{Bin}\left(n_0, p(\rho - \lfloor\rho\rfloor)/b\right), \end{aligned}$$

where $b = n/\log^2 n$ and

$$\rho = \frac{32\ln(4/\delta)}{\varepsilon^2} \cdot \frac{b}{n/2r} = O(1).$$

Recall that $p_{\mathrm{col}} < 1/b$ is the probability of hash collision. Therefore, $\mathbb{E}[X] \leq p(g_i(x) + n_0 p_{\mathrm{col}} + n_0\rho/b) = pg_i(x) + O(pn_0/b)$. Assume $g_i(x) < \phi n_0/8$. Since $\phi = \Omega(\log^2 n/n) = \Omega(1/b)$, there exists large enough n such that $O(pn_0/b) < p\phi n_0/24$. Therefore, $\mathbb{E}[X] \leq p\phi n_0/8 + p\phi n_0/24 = p\phi n_0/6 = 2\Delta/3$. Let $c = \Delta/\mathbb{E}[X] - 1 \geq 1/2$. By Chernoff bound,

$$\begin{aligned} \Pr[X \geq \Delta] &= \Pr[X \geq (1+c)\mathbb{E}[X]] \leq \exp\left(-\mathbb{E}[X] \cdot \frac{c^2}{2+c}\right) \\ &= \exp\left(-\frac{\Delta c^2}{(2+c)(1+c)}\right) \leq \exp\left(-\frac{\Delta}{15}\right) \leq \frac{1}{\phi B}. \end{aligned}$$

Note that the number of elements x with $g_i(x) \geq \phi n_0/8$ is at most $8/\phi$, and $|C_i| \leq B$. Therefore, we have

$$\begin{aligned} \mathbb{E}[|C_i|] &\leq \Pr[n_i > n_0] \cdot \mathbb{E}[|C_i| \mid n_i > n_0] + \mathbb{E}[|C_i| \mid n_i \leq n_0] \\ &\leq O(1/Bn) \cdot B + (1/\phi B) \cdot B + 8/\phi = O(1/\phi) \end{aligned}$$

Next we prove (2). Each user chooses exactly one level, and in this level, each user is expected to send $p(1+\rho)$ messages where $\rho = O(1)$, so the total number of messages each user sends is $O(p)$.

Finally we prove (3). The analyzer runs in three steps:

1. Classify the messages to the corresponding levels. The cost is linear to the number of total messages, which is $O(pn)$.
2. Estimate the occurrences of all elements in $[2^s]$. By Theorem 3.7, the cost is
$$O(p \cdot 2^s(\mathbb{E}[n_s]/b + \log(1/\delta)/\varepsilon^2)) = O(pn).$$
3. Filter the set of candidates and extend it to the next level. For the i-th level, note that
$$\begin{aligned} \mathbb{E}[|C_i| \cdot n_i] &\leq \Pr[n_i > n_0] \cdot Bn + n_0 \cdot \mathbb{E}[|C_i| \mid n_i \leq n_0] \\ &\leq O(n_0/\phi). \end{aligned}$$
Therefore, the expected cost is
$$O\left(\sum_{i=s}^{t} p\mathbb{E}[|C_i| \cdot n_i]\right) = O(pn/\phi).$$

Summing up all costs, the expected total running time is $O(pn/\phi)$. □

Theorem 4.4. *Assume $\log n = \Theta(\log B) = \Theta(\log(1/\delta)) = \Theta(\log(1/\beta))$, $\phi = \Omega(\log^2 n/n)$, and ε is a constant. $\mathcal{P}^{\mathrm{HHD}}$ a private-coin (ε,δ)-shuffle DP protocol, in which each user sends $O(\log^2 n/\phi n)$ messages in expectation, each consisting of $O(\log B)$ bits. All the heavy hitters are reported in $O(\log^2 n/\phi^2)$ time with probability at least $1-\beta$.*

Proof. We set

$$p = \frac{30r}{\phi n} \ln\left(\frac{r}{\phi\beta} + \phi B\right) = \Theta\left(\frac{\log^2 n}{\phi n}\right),$$

which is less than 1 for $\phi = \Omega(\log^2 n/n)$. Then p satisfies both the requirements of Lemma 4.2 and 4.3, and we conclude this theorem by Lemma 4.1–4.3. □

By also running our frequency estimation protocol (using half of the privacy budget, say), we can estimate the frequencies of all elements with $1 + o(1)$ messages per user in total time $O(n^2)$, for any $B = n^{O(1)}$:

Corollary 4.5. *For any $B = n^{O(1)}$ and any constant $c > 2$, there is a $1 + o(1)$-message (ε,δ)-shuffle DP protocol that estimate the frequency of all elements with $O(\log^c n)$ error in time $O(n^2/\log^c n)$.*

Proof. Take $\phi = \log^c n/n$ in $\mathcal{P}^{\mathrm{HHD}}$ and $b = n/\log^3 n$ in $\mathcal{P}^{\mathrm{FE1}}$. Then the number of messages per user is $1 + O(1/\log^2 n) + O(\log^2 n/\phi n) = 1 + o(1)$. The frequencies of all the heavy hitters are given by $\mathcal{A}^{\mathrm{FE1}}$ with error $O(\log^2 n)$. The frequencies of all the light hitters are estimated as 0, so the error is $O(\phi n) = O(\log^c n)$. The running time of $\mathcal{A}^{\mathrm{HHD}}$ and $\mathcal{A}^{\mathrm{FE1}}$ are $O(\log^2 n/\phi^2)$ and $O(n/\phi)$ respectively, so the total query time is $O(n^2/\log^c n)$. □

5 1-SPARSE VECTOR SUMMATION

5.1 Review of Ghazi et al. [27]

Recall that in the real summation problem, the i-th user holds a real number $w_i \in [0,1]$, and the goal is to estimate their sum. Let $\Delta \geq 1$ be a quantization parameter. The shuffle-DP protocol of [27] sends the following messages:

(1) sends $\lfloor \Delta w_i \rfloor + \mathrm{Ber}(\Delta w_i - \lfloor \Delta w_i \rfloor)$ if this value is nonzero;
(2) sends the central-DP noise, which is nonzero with probability $\tilde{O}(\Delta/n)$ (they show that the sum of these noises almost matches the discrete Laplace distribution); and
(3) sends a random multiset S of zero-sum noises to ensure shuffle-DP, where each $s \in S$ is taken from $\{-\Delta, \ldots, \Delta\} - \{0\}$ and $\mathbb{E}[|S|] = \tilde{O}(\Delta/n)$.

The analyzer simply adds up all messages received and scales it down by Δ. The estimator is unbiased with variance $O(1 + \sum_i w_i/\Delta^2)$, where $O(\sum_i w_i/\Delta^2)$ comes from (1) and $O(1)$ comes from (2).

For the 1-sparse vector summation problem, the i-th user holds an element $x_i \in [B]$ together with a weight $w_i \in [0,1]$, and the goal is to obtain an estimate $\hat{g}_x$ for $g_x = \sum_{i=1}^n w_i \cdot \mathbb{I}[x_i = x]$ for every $x \in [B]$ so as to minimize the MSE $= \sum_x (\hat{g}_x - g_x)^2$. Ghazi et al. [27] simply apply their real summation protocol on each coordinate $x \in [B]$, each with $(\varepsilon/2, \delta/2)$ privacy budget, resulting in a message number of $1 + \tilde{O}(\Delta B/n)$ with MSE $O(B + \sum_i w_i/\Delta^2) = O(B + n/\Delta^2)$.

Ghazi et al. [27] set $\Delta = \sqrt{n}$ so the variance on each coordinate is $O(1)$. However, we observe that setting $\Delta = \sqrt{n/B}$ does not increase the MSE asymptotically (it is still $O(B)$), while the number of messages can be reduced to $1 + \tilde{O}(\sqrt{B/n})$, which is $1 + o(1)$ when $B = o(n)$. Thus, they can achieve the optimal central-DP MSE with $1 + o(1)$ messages for the low-dimensional setting $B = o(n)$.

In the high-dimensional setting $B = \Omega(n)$, the optimal central-DP MSE is $\tilde{O}(n)$, by simply running the Laplace mechanism followed by setting all estimates less than $\log B/\varepsilon$ to 0. We note that this "zeroing" post-processing step can also be applied to [27], so that their MSE can be reduced to $\tilde{O}(n)$ when $B > n$. However, their message number is at least $\tilde{\Omega}(B/n)$ even if we set $\Delta = 1$.

5.2 Our protocol

Below, we show how to achieve $\tilde{O}(n)$ MSE with $1 + o(1)$ messages in the high-dimensional setting, based on our frequency estimation and heavy hitter detection protocols.

User i with input (x_i, w_i) applies randomized rounding to w_i. More precisely, we extend the domain to $[B+1]$ with a dummy element B, and then convert x_i to $f(x_i, w_i)$ where

$$f(x,w) = \begin{cases} x, & \text{with probability } w; \\ B, & \text{otherwise.} \end{cases}$$

Then we just apply the protocol from Corollary 4.5, namely, heavy hitter detection + frequency estimation.

Theorem 5.1. *When $B = \Omega(n)$, there is a (ε, δ)-shuffle DP protocol for the 1-sparse vector summation problem that sends $1 + o(1)$ messages while achieving MSE $O(n \log^c n)$ for any $c > 2$. The analyzer's running time is $O(n^2/\log^c n)$.*

Proof. Since for any two neighboring inputs, after applying f, the outputs are also neighboring if not identical, so the privacy of the protocol still holds. The message number and running time directly follow Corollary 4.5. Below we analyze the MSE.

Let H be the set of heavy hitter candidates, then given $x \in H$, the frequency of x is estimated by our frequency estimation protocol. By the proof of Lemma 3.6, our estimator is

$$\hat{g}_x = (X - n\rho/b - np_{\mathrm{col}})/(1 - p_{\mathrm{col}}),$$

where

$$X = \left(\sum_{i=1}^n X_i\right) + \mathrm{Bin}(n\lfloor\rho\rfloor, 1/b) + \mathrm{Bin}(n, (\rho - \lfloor\rho\rfloor)/b),$$

$b = n/\log^2 n$, $X_i = z_{xi} + (1 - z_{xi})\mathrm{Ber}(p_{\mathrm{col}})$, and $z_{xi} = \mathbb{I}[f(x_i, w_i) = x]$. Since $X_i^2 = X_i$, we have $\mathrm{Var}[X_i] = \mathbb{E}[X_i] - \mathbb{E}[X_i^2] \leq \mathbb{E}[X_i]$, while binomial distributions have the similar inequality. Since $\mathbb{E}[\hat{g}_x] = g_x$, and X_i, X_j are independent for all $i \neq j$, we have

$$\begin{aligned} \mathbb{E}[(\hat{g}_x - g_x)^2 \mid x \in H] &= \mathrm{Var}(\hat{g}_x \mid x \in H) \\ &\leq \mathbb{E}[X]/(1 - p_{\mathrm{col}})^2 \\ &\leq (g_x + (n - g_x)p_{\mathrm{col}} + n\rho/b)/(1 - p_{\mathrm{col}})^2 \\ &= O(g_x + \log^2 n). \end{aligned}$$

Let ϕ be the threshold chosen in Corollary 4.5, then $\phi n = \log^c n = \omega(\log^2 n)$. We set $\beta = 1/n$ in Lemma 4.2, then with probability at least $1 - 1/n$, all x with $\bar{g}_x \geq \phi n$ are in the candidate set, where $\bar{g}_x = \sum_{i=1}^n z_{xi}$. On the other hand, by Chernoff bound, $\Pr[\bar{g}_x < g_x/2] = \exp(-\Omega(g_x))$. When $g_x \geq 2\phi n = \omega(\log^2 n)$, it is $\Pr[\bar{g}_x < \phi n] = \exp(-\omega(\log^2 n)) = O(1/n)$, which implies $\Pr[x \notin H] \leq \Pr[\bar{g}_x < \phi n] + \Pr[x \notin H \mid \bar{g}_x \geq \phi n] = O(1/n)$. For any $x \notin H$, our estimator is simply $\hat{g}_x = 0$. Therefore,

$$\begin{aligned} \mathrm{MSE} &= \sum_{x \in [B]} \mathbb{E}[(\hat{g}_x - g_x)^2] \\ &\leq \sum_{x: g_x \geq 2\phi n} (\mathrm{Var}(\hat{g}_x \mid x \in H] + g_x^2 \cdot \Pr[x \notin H]) + \sum_{x: g_x < 2\phi n} g_x^2 \\ &= O\left(\sum_{x \in [B]} g(x) + \frac{\log^2 n}{\phi} + \frac{1}{n}\sum_{x \in [B]} g_x^2 + \sum_{x: g_x \leq 2\phi n} g_x^2\right) \\ &= O\left(n + n + n + 2\phi n \sum_{x \in [B]} g_x\right) \\ &= O(n \log^c n). \end{aligned}$$

□

Remark 1. Instead of using heavy hitter detection + frequency estimation, we could also use the frequency estimation protocol alone (followed by the zeroing post-processing step), which can improve the MSE by a polylogarithmic factor but at the expense of a running time linear in B.

Remark 2. Our protocol can be easily extended to the case where each user has a k-sparse vector. For *coordinate-level* DP (i.e., neighboring datasets differ by one coordinate in some user's vector), we can treat each of the k coordinates as a 1-sparse vector, which increases the message number by a factor of k; for *vector-level* DP (i.e., neighboring datasets may differ by the whole vector owned

by some user), we can use group privacy [38] to scale (ε, δ) down to $(\varepsilon/k, \delta/ke^{\varepsilon})$ to reduce this case to coordinate-level DP.

6 EXPERIMENTS

We have implemented our protocols and the protocols in [25, 27] in C++[4], and conducted all the experiments on a server with an Intel Xeon Silver 4116 CPU. Note that, however, the accuracy and message number of the protocols do not depend on the machine, only the running time of the analyzer does. The randomizer's time is negligible.

We used both real and synthetic data in the experiments:

- The AOL dataset [33] is a collection of real-world website accesses. We truncate each URL to a prefix of a certain length, so as to obtain different domain sizes. For example, truncating to the first 3 characters generates a domain of size $B = 2^{24}$, since each character has 8 bits.
- We used the Zipfian distribution to generate synthetic data of varying levels of skewness. We first randomly permute the elements in $[B]$. Then we draw elements following the distribution Zipf(α, B), whose probability mass for the k-th element in the permutation is $k^{-\alpha}/\sum_{i=1}^{B} i^{-\alpha}$, so a larger α corresponds to more skew.

6.1 Frequency Estimation

We evaluate our large-domain protocol against the start-of-the-art method [25], denoted as GGKPV. We compare them along the following metrics: (1) error, including the maximum error, the 95% error, the 90% error, and the median error over all elements in $[B]$; (2) the expected number of messages sent per user; (3) query time to estimate the frequency of a single element; and (4) the query time to estimate all frequencies.

The results on the AOL dataset are shown in Figure 1. Note that the error is in linear scale while the message number is shown in log scale. Recall that the parameter c controls the error-message number trade-off of our protocol. In particular, $c = 1$ has $O(\log n)$ error (same as that of GGKPV) with $O(1)$ messages (GGKPV has $O(\log n)$ message). The experimental results suggest that our actual error is smaller than half that of GGKPV, suggesting that the hidden constant in the $O(\log n)$ error in our protocol is smaller. The message number is 1/100 that of GGKPV, which is also more than what the theory predicts. Part of the reason is the parameter optimization (see the remark after Theorem 3.1), which reduces the message number by 70%–90%. Choosing a larger c tunes the trade-off: With $c = 3$, each user sends $1 + o(1)$ messages, albeit at the expense of an $O(\log^2 n)$ error.

Table 2 shows the results on different ε. GGKPV only supports $\varepsilon \le 1$, while for very small ε, it cannot finish within 1 day. From the results, we see that, a larger ε leads to smaller error, less messages, and faster query time, which are as expected. The more than 100x reduction of query time is also very significant and useful in practice. Table 3 shows results for different n, the number of users. As the theory predicts, message number, the error, and the query time of all elements of our protocol are logarithmically affected by n, while the query time of a single element is linear to n.

[4]https://github.com/hkustDB/SDPFE

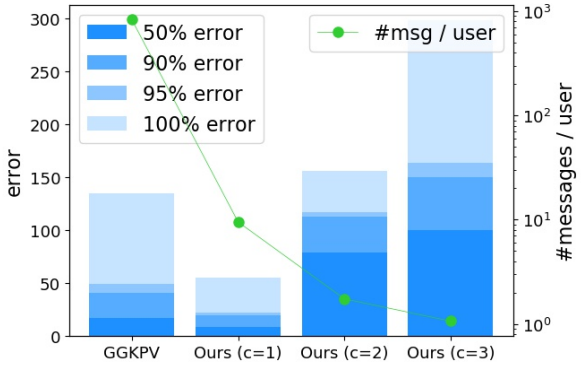

Figure 1: Comparison among frequency estimation protocols on error and #messages/user ($n = 10^5, B = 2^{24}, \varepsilon = 1, \delta = n^{-2}$)

6.2 Heavy Hitter Detection

Table 4 compares different protocols for heavy hitter detection. We use a large domain $B = 2^{48}$. For a smaller B, we could just use our frequency estimation protocol, but for such a large B, it would take more than 1 year (estimated) for it to obtain all frequencies. Although the message number of GGKPV-HHD [25] is polylogarithmic, it is gigantic number in practice. On the other hand, our HHD protocol provides a highly-efficient and light-weight solution to this problem. The recall rate is 100% in all our experiments, namely, all heavy hitters have been identified. There are false positives, though, but they can be removed by querying the frequency estimation data structure, subject to the error in the estimated frequencies. The costs of our HHD also depend on ϕ, where a larger ϕ reduces all costs, as predicted by the theoretical analysis.

6.3 Sparse Vector Summation

To conduct experiments on the 1-sparse vector summation problem, we added a weight to each user's element. For the AOL dataset, each user reports its most visited URL and the weight is the number of times visited normalized to $(0, 1)$. For Zipfian data, we added uniform random numbers in $(0, 1)$ as weights.

As described in Section 5.2, we can solve the 1-sparse vector summation problem by combining heavy hitter detection and frequency estimation (denoted as HHD+FE) for very large B, or by frequency estimation alone for $n < B < \tilde{O}(n^2)$ (denoted FE; see Remark 1). We compare them with the protocol in [27] (denoted as GKMPS), which simply executes B instances of real summation with $\Delta = 1$. Each real summation generates a discrete Laplace noise with scale $2/(1-\gamma)\varepsilon$, where γ is a parameter that controls the trade-off between error and message number. We set $\gamma = 0.9$ following [27]. Directly doing so results in $\tilde{O}(B)$ MSE (denoted as "DLap"), while applying zeroing post-processing step as described in Section 5.1, which reduces the MSE to $\tilde{O}(n)$. Our experiments demonstrate that this zeroing post-processing step significantly reduces their MSE in practice. We do the same zeroing post-processing step for HHD+FE and FE as well.

We first compare all the protocols on the AOL dataset with B slightly larger than n, and the results are shown in Table 5. Although

	95% error						#Messages / user					
ε	0.25	0.5	0.75	1	2	3	0.25	0.5	0.75	1	2	3
GGKPV	-	98.68	65.67	49.24	-	-	-	3317	1475	830	-	-
Ours ($c = 1$)	72.99	38.94	25.83	22.47	13.68	12.88	114.82	30.65	15.010	9.51	4.23	3.29
Ours ($c = 3$)	1395.4	411.21	228.28	163.94	102.20	92.24	1.85	1.22	1.10	1.063	1.024	1.017
	Query time (all / s)						Query time (single /s)					
ε	0.25	0.5	0.75	1	2	3	0.25	0.5	0.75	1	2	3
GGKPV	> 1 day	99531.8	42581.2	23908.6	-	-	-	48.44	16.45	13.11	-	-
Ours ($c = 1$)	956.32	231.51	118.27	70.91	31.69	23.91	0.359	0.095	0.047	0.028	0.013	0.009
Ours ($c = 3$)	1106.27	1140.04	1182.6	1047.19	1040.12	1151.4	0.003	0.003	0.003	0.003	0.003	0.003

Table 2: Comparison among frequency estimation protocols varying privacy parameter ε ($n = 10^5, B = 2^{24}, \delta = 10^{-10}, b = 10^5/16.6^c$)

	95% error				#Messages / user			
n	10^4	10^5	10^6	10^7	10^4	10^5	10^6	10^7
GGKPV	39.12	49.24	60.14	-	665	830	996	-
Ours ($c = 1$)	18.40	22.47	27.32	29.64	9.265	9.51	9.65	9.75
Ours ($c = 3$)	132.55	163.94	204.57	247.23	1.09	1.064	1.045	1.034
	Query time (all / s)				Query time (single / s)			
n	10^4	10^5	10^6	10^7	10^4	10^5	10^6	10^7
GGKPV	11702.9	23908.6	53752.6	> 1 day	0.55	16.43	225.41	-
Ours ($c = 1$)	57.1	70.91	98.22	139.47	0.0028	0.0282	0.2836	2.9131
Ours ($c = 3$)	545.43	1047.19	1839.29	2859.06	0.0003	0.0031	0.0312	0.3119

Table 3: Comparison among frequency estimation protocols varying number of users n ($B = 2^{24}, \varepsilon = 1, \delta = n^{-2}, b = n/\log^c n$)

Protocols	Our HHD				GGKPV-HHD	Our FE (c=3)
Threshold ϕ	0.01	0.0075	0.005	0.0025	-	-
#Messages / user	0.21	0.28	0.40	0.77	$> 10^8$	1.034
Message size / user (bytes)	3.57	4.95	7.08	13.85	> 20 GB	18.61
Query time (s)	41.29	190.45	612.85	4802.28	> 1 year	> 1 year
Recall rate	100%	100%	100%	100%	100%	100%

Table 4: Comparison among heavy hitter detection protocols ($n = 10^7, B = 2^{48}, \varepsilon = 1, \delta = 10^{-14}$)

Mechanisms		# Messages / user	MSE ($\times 10^8$)	Query time (s)
GKMPS	DLap	16788.66	134.11	1600
	Zeroing		0.65	
HHD+FE		1.15	4.30	548
FE	$c = 1$	9.75	0.09	101
	$c = 2$	1.54	0.67	109
	$c = 3$	1.03	1.54	1349

Table 5: Comparison among vector summation protocols ($n = 10^7, B = 2^{24}, \varepsilon = 1, \delta = 10^{-14}, \phi = 7 \times 10^{-4}$)

Metrics	# Messages / user		MSE ($\times 10^8$)			Query time (s)	
Mechanisms	HHD+FE	GKMPS	HHD+FE	GKMPS		HHD+FE	GKMPS
				DLap	Zeroing		
AOL	1.14	$> 10^6$	18.01	8590	3.69	1724	> 1 day
Zipf ($\alpha = 1$)			5.50		1.07		
Zipf ($\alpha = 2$)			0.42		0.06		
Zipf ($\alpha = 3$)			0.08		0.01		

Table 6: Comparison among vector summation protocols varying datasets ($n = 10^7, B = 2^{30}, \varepsilon = 1, \delta = 10^{-14}, \phi = 7 \times 10^{-4}$)

theoretically GKMPS sends $1 + \tilde{O}(B/n)$ messages, this number in the experiments is more than 10,000, indicating an impractical hidden polylogarithmic factor. At the same time, our protocols send much fewer messages while achieving a similar or smaller error. By taking $c = 3$, both HHD+FE and FE send $1 + o(1)$ messages while achieving a near-optimal MSE. Setting a smaller c reduces the MSE to be smaller than that of GKMPS while the message number increases to a small constant. Comparing our two protocols, FE outperforms HHD+FE, because the later assigns half of the privacy budget to detect the heavy coordinates.

Next, we tested with a $B \gg n$ on the AOL dataset and the Zipfian datasets and the results are shown in Table 6. For such a large B, neither FE nor GKMPS can finish in a day. The MSE of GKMPS is computed by assuming no error over all the 0 coordinates, which yields a (small) underestimate of their actual MSE. The results on the Zipfian datasets show skewness affects the performance. We observe that the number of messages and the query time of HHD+FE do not really change, but the MSE of both HHD+FE and GKMPS (with zeroing) decreases on more skewed data. This is because as skewness increases, the MSE from those coordinates less than the zeroing threshold also reduces.

7 OPEN PROBLEMS

Results in this paper have further improved the power of shuffle-DP with just $1 + o(1)$ messages per user. A major open problem is thus: are there any natural problems for which the optimal central-DP error is not achievable (subject to polylogarithmic factors) with $1 + o(1)$ messages?

There are also some interesting but more technical questions that are still open: (1) For frequency estimation, our $1+o(1)$ protocol still does not match the optimal central-DP error of $O(\log n)$, but with an $\omega(1)$ gap. Closing this gap could be an interesting theoretical problem. (2) For k-sparse vector summation under vector-level DP, the use of group privacy perhaps does not yield the best dependency on k. We note that this problem is equivalent to (the sparse case of) mean estimation in high dimensions, and techniques from [9, 29, 40] might be useful for further improvement.

ACKNOWLEDGMENTS

This work has been supported by HKRGC under grants 16201819, 16205420, and 16205422. We thank the anonymous reviewers of CCS '22 for valuable suggestions on improving the presentation of the paper.

REFERENCES

[1] Apple Differential Privacy Team. [n.d.]. Apple Differential Privacy Technical Overview. https://www.apple.com/privacy/docs/Differential_Privacy_Overview.pdf.
[2] Victor Balcer and Albert Cheu. 2020. Separating Local & Shuffled Differential Privacy via Histograms. In *1st Conference on Information-Theoretic Cryptography*. 1:1–1:14.
[3] Victor Balcer, Albert Cheu, Matthew Joseph, and Jieming Mao. 2021. Connecting Robust Shuffle Privacy and Pan-Privacy. In *Proceedings of the Thirty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*. 2384–2403.
[4] Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. 2019. The Privacy Blanket of the Shuffle Model. In *CRYPTO*.
[5] Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. 2020. Private Summation in the Multi-Message Shuffle Model. In *ACM SIGSAC Conference on Computer and Communications Security*.
[6] Raef Bassily, Kobbi Nissim, Uri Stemmer, and Abhradeep Thakurta. 2017. Practical Locally Private Heavy Hitters. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 2285–2293.
[7] Raef Bassily and Adam Smith. 2015. Local, private, efficient protocols for succinct histograms. In *ACM STOC*. 127–135.
[8] Amos Beimel, Iftach Haitner, Kobbi Nissim, and Uri Stemmer. 2020. On the Round Complexity of the Shuffle Model. In *TCC 2020*. https://arxiv.org/abs/2009.13510
[9] Sourav Biswas, Yihe Dong, Gautam Kamath, and Jonathan Ullman. 2020. CoinPress: Practical Private Mean and Covariance Estimation. In *Advances in Neural and Information Processing Systems*.
[10] Andrea Bittau, Úlfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnes, and Bernhard Seefeld. 2017. Prochlo: Strong Privacy for Analytics in the Crowd. In *Proceedings of the 26th Symposium on Operating Systems Principles*. 441–459.
[11] Mark Bun, Kobbi Nissim, and Uri Stemmer. 2016. Simultaneous Private Learning of Multiple Concepts. In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science*. 369–380.
[12] T-H. Hubert Chan, Elaine Shi, and Dawn Song. 2012. Optimal Lower Bound for Differentially Private Multi-Party Aggregation. In *Proceedings of the 20th Annual European Conference on Algorithms*. 277–288.
[13] David L. Chaum. 1981. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Commun. ACM* 24 (1981), 84–90.
[14] Lijie Chen, Badih Ghazi, Ravi Kumar, and Pasin Manurangsi. 2021. On Distributed Differential Privacy and Counting Distinct Elements. In *ITCS*. 56:1–56:18.
[15] Albert Cheu, Adam Smith, Jonathan Ullman, David Zeber, and Maxim Zhilyaev. 2019. Distributed differential privacy via shuffling. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. 375–403.
[16] Albert Cheu and Maxim Zhilyaev. 2021. Differentially Private Histograms in the Shuffle Model from Fake Users. *CoRR* (2021).
[17] G. Cormode and M. Hadjieleftheriou. 2010. Methods for finding frequent items in data streams. *The VLDB Journal* 19, 1 (2010), 3–20.
[18] Ivan Damgård, Jesper Buus Nielsen, Rafail Ostrovsky, and Adi Rosén. 2016. Unconditionally Secure Computation with Reduced Interaction. In *Proceedings, Part II, of the 35th Annual International Conference on Advances in Cryptology — EUROCRYPT 2016 - Volume 9666*. Springer-Verlag, Berlin, Heidelberg, 420–447.
[19] G. Danezis, R. Dingledine, and N. Mathewson. 2003. Mixminion: design of a type III anonymous remailer protocol. In *Symposium on Security and Privacy*. 2–15.
[20] Roger Dingledine, Nick Mathewson, and Paul Syverson. 2004. Tor: The Second-Generation Onion Router. In *13th USENIX Security Symposium (USENIX Security 04)*.
[21] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. 2019. Amplification by shuffling: From local to central differential privacy via anonymity. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2468–2479.
[22] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response. In *ACM SIGSAC Conference on Computer and Communications Security*. ACM, 1054–1067. https://doi.org/10.1145/2660267.2660348
[23] Vitaly Feldman and Kunal Talwar. 2021. Lossless Compression of Efficient Private Local Randomizers. In *International Conference on Machine Learning*.
[24] Jen Fitzpatrick and Karen DeSalvo. [n.d.]. Helping public health officials combat COVID-19. https://blog.google/technology/health/covid-19-community-mobility-reports/.
[25] Badih Ghazi, Noah Golowich, Ravi Kumar, Rasmus Pagh, and Ameya Velingker. 2021. On the Power of Multiple Anonymous Messages: Frequency Estimation and Selection in the Shuffle Model of Differential Privacy. In *Advances in Cryptology – EUROCRYPT 2021*. 463–488.
[26] Badih Ghazi, Ravi Kumar, Pasin Manurangsi, and Rasmus Pagh. 2020. Private Counting from Anonymous Messages: Near-Optimal Accuracy with Vanishing Communication Overhead. In *Proceedings of the 37th International Conference on Machine Learning*.
[27] Badih Ghazi, Ravi Kumar, Pasin Manurangsi, Rasmus Pagh, and Amer Sinha. 2021. Differentially Private Aggregation in the Shuffle Model: Almost Central Accuracy in Almost a Single Message. In *International Conference on Machine Learning*.
[28] Xi He, Ashwin Machanavajjhala, Cheryl Flynn, and Divesh Srivastava. 2017. Composing differential privacy and secure computation: A case study on scaling private record linkage. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. 1389–1406.
[29] Ziyue Huang, Yuting Liang, and Ke Yi. 2021. Instance-optimal Mean Estimation Under Differential Privacy. In *Advances in Neural Information Processing Systems*.
[30] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai. 2006. Cryptography from Anonymity. In *FOCS*.
[31] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. 2015. Secure multi-party differential privacy. *Advances in neural information processing systems* 28 (2015).
[32] Andreas Kopp. [n.d.]. Microsoft SmartNoise Differential Privacy Machine Learning Case Studies. https://azure.microsoft.com/en-in/resources/microsoft-

smartnoisedifferential-privacy-machine-learning-case-studies/.
[33] Greg Pass, Abdur Chowdhury, and Cayley Torgeson. 2006. A picture of search. In *Proceedings of the 1st international conference on Scalable information systems*.
[34] Martin Pettai and Peeter Laud. 2015. Combining differential privacy and secure multiparty computation. In *Proceedings of the 31st Annual Computer Security Applications Conference*. 421–430.
[35] M.G. Reed, P.F. Syverson, and D.M. Goldschlag. 1998. Anonymous connections and onion routing. *IEEE Journal on Selected Areas in Communications* (1998), 482–494.
[36] Michael K. Reiter and Aviel D. Rubin. 1998. Crowds: Anonymity for Web Transactions. *ACM Transactions on Information and System Security* (1998), 66–92.
[37] Elaine Shi, T.-H. Hubert Chan, Eleanor Rieffel, and Dawn Song. 2017. Distributed Private Data Analysis: Lower Bounds and Practical Constructions. *ACM Transactions on Algorithms* (2017).
[38] Salil Vadhan. 2017. *The Complexity of Differential Privacy*. 347–450.
[39] Tianhao Wang, Jeremiah Blocki, Ninghui Li, and Somesh Jha. 2017. Locally differentially private protocols for frequency estimation. In *26th USENIX Security Symposium Security*. 729–745.
[40] Mingxun Zhou, Tianhao Wang, T-H. Hubert Chan, Giulia Fanti, and Elaine Shi. 2022. Locally Differentially Private Sparse Vector Aggregation. In *2022 IEEE Symposium on Security and Privacy (SP)*. 422–439. https://doi.org/10.1109/SP46214.2022.9833635
[41] Wennan Zhu, Peter Kairouz, Brendan McMahan, Haicheng Sun, and Vivian (Wei) Li. 2020. Federated Heavy Hitters with Differential Privacy. In *International Conference on Artificial Intelligence and Statistics (AISTATS) 2020*.