Differential Privacy via Group Shuffling

Amir Mohammad Abouei
Department of Computer Engineering
Sharif University of Technology
amabouei@ce.sharif.edu

Clément L. Canonne
School of Computer Science
University of Sydney
clement.canonne@sydney.edu.au

Abstract

The past decade has seen data privacy emerge as a fundamental and pressing issue. Among the tools developed to tackle it, differential privacy has emerged as a central and principled framework, with specific variants capturing various threat models. In particular, the recently proposed shuffle model of differential privacy allows for promising tradeoffs between accuracy and privacy. However, the shuffle model may not be suitable in all situations, as it relies on a distributed setting where all users can coordinate and trust (or simulate) a joint shuffling algorithm.

To address this, we introduce a new model, the *group shuffle* model, in which users are partitioned into several groups, each group having its own local shuffler. We investigate the privacy/accuracy tradeoffs in our model, by comparing it to both the shuffle and local models of privacy, which it some sense interpolates between. In addition to general relations between group shuffle, shuffle, and local privacy, we provide a detailed comparison of the cost and benefit of the group shuffle model, by providing both upper and lower bounds for the specific task of binary summation.

1 Introduction

While distributed machine learning and, more generally, distributed computing have been around for a long time, they recently have picked up significant speed, with the need to process and learn from distributed data becoming pervasive. Privacy considerations have also become front and center, due to the need to process sensitive or personal user data.

Differential privacy (DP) [8] has emerged as principled way to address the latter aspect, by providing a mathematical framework to quantify and guarantee the privacy provided by a given algorithm. Several variants of DP have been proposed, each addressing a different "threat" model – which parties involved can be trusted with the data, and which must be protected against – and as a result each coming with different trade-offs between privacy and utility.

In the central model of DP (often simply referred to as DP), all users trust a central party, known as the curator or analyzer, with their data. The curator then runs an algorithm on this raw data, and the output of this algorithm is the quantity which is then required to reveal little about the original inputs. This privacy requirement allows for quite good utility, but does not protect against privacy leaks by the curator itself. On the other side of the spectrum, in the local model of DP (LDP) [13, 12, 7], each user privatises their data before sending it to the central entity, thus guaranteeing privacy against all parties involved. The resulting privacy guarantee is very strong, but comes at a cost of much lower utility achievable.

An intermediate setting is the recently introduced *shuffle model* of privacy [6, 3], which only requires users to trust a third party, the "shuffler," which upon receiving their messages applies a uniformly random permutation before delivering the shuffled result to the analyzer. This trusted third party, in view of the simple task it performs, can be efficiently implemented using cryptographic primitives (e.g., multiparty computation (MPC)); yet, by only requiring the shuffled messages to provide privacy

Preprint. Under review.

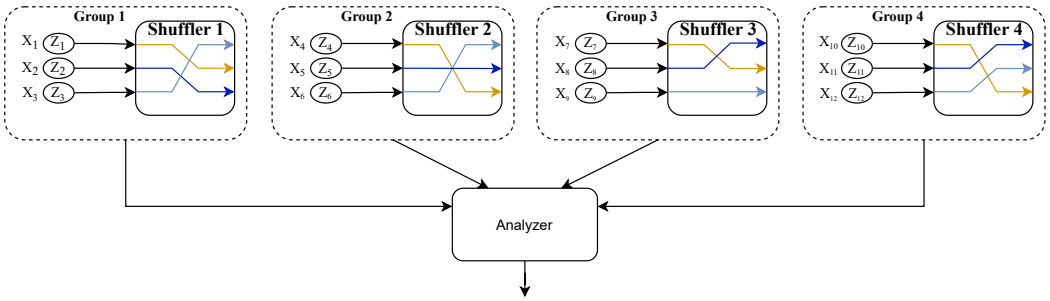

Figure 1: The proposed Group Shuffle model with 12 users and 4 Groups. X_i's and Z_i's are inputs of users and outputs of randomizers respectively, where each group has a independent shuffler.

guarantees instead of the original set of messages, shuffle privacy allows for much better utility than LDP, sometimes even comparable to that achievable in the central DP model.

Unfortunately, even this seemingly minimal requirement (a common, trusted shuffler) can be hard or impossible to achieve. This could be the case where users are distributed across disparate entities, such as different companies, or even countries, each with incompatible infrastructures or requirements. Coordination across those various entities to agree on or simulate a common shuffler could be technically infeasible, legally impossible, or politically complicated; still, each of those entities could quite easily implement a shuffler of their own. Addressing this type of scenario calls for a new privacy model, the *group shuffle* model, which we introduce below (see Section 2 for the formal definition).

In the group shuffle model, users are partitioned into a fixed number of groups, and to each group corresponds a different (independent across groups) shuffler. Doing so has three immediate consequences: first, this removes the need for the different entities (corresponding to the groups) to agree on a single trusted third party. Second, by having a much smaller number of inputs to each shuffler, one can hope for an easier implementation, and better efficiency overall. Finally, this provides better robustness: even if one or some of the shufflers were to be compromised by users behaving adversarially or security breaches on the shufflers, only the privacy of the corresponding fraction of users could be jeopardised.

Organization. We formally define our new model in Section 2, before comparing and relating it to the local and shuffle models in Section 3, where we provide general results and reductions between the three. We then provide in Section 4, as an important and illustrative example, an extensive study of what can be achieved in the group shuffle model for the problem of binary summation. Finally, we conclude in Section 5 by discussing some aspects of our model and directions for future work.

2 Our model

We now introduce our privacy model, after setting up the required notation. Hereafter, we consider a setting with n users, each of them with their own datum, which takes values in $\mathcal{X}$.

A protocol $\mathcal{P} = (\mathcal{R}, \mathcal{S}, \mathcal{A}, k)$ in our model, analogously to the shuffle model, is comprised of 3 algorithms:

- A *randomizer* $\mathcal{R}\colon \mathcal{X} \times \{0,1\}^r \to \mathcal{Y}^*$. Each user has a randomizer which converts an input and a public random bits to m-message. The number of messages can be variable; i.e., m itself can be a random variable.
- k *shufflers*, each an instance of the same $S\colon \mathcal{Y}^* \to \mathcal{Y}^*$. S takes a (multi)set of messages and uniformly permutes these messages.
- An *analyzer* $\mathcal{A}\colon \mathcal{Y}^* \times \{0,1\}^r \to \mathcal{Z}$. The analyzer gets as input an (unordered) tuple of messages with a public random bits, and processes them in order to estimate or compute a function $f(x_1, x_2, \ldots, x_n)$ of the n inputs.

In the above, the parameter k denotes the number of groups. In contrast to the usual shuffle model, which includes a single shuffler, in the group model, there are k independent shufflers, permuting the messages of disjoint subsets of the n users. For simplicity, we hereafter assume that n is divisible by k, and that the n users are partitioned into k groups of $n' := \frac{n}{k}$ users. Hence, the output of the

protocol is

$$\mathcal{P}(\vec{x}) = \mathcal{A}(S^k \circ \mathcal{R}^n(\vec{x}, W), W) \tag{1}$$

where $\vec{x} = (x_1, \ldots, x_n)$ is dataset of n inputs to the users, W is a public random string, and we define $S^k \circ \mathcal{R}^n(\vec{x}, W)$ by

$$(S_1(\mathcal{R}(x_1, W), \ldots, \mathcal{R}(x_{n'}, W)), \ldots, S_k(\mathcal{R}(x_{(k-1)n'+1}, W), \ldots, \mathcal{R}(x_{kn'}, W)). \tag{2}$$

Definition 1 (Group Shuffle private)**.** *Let $\varepsilon > 0$ and $\delta \in [0, 1]$. A protocol $\mathcal{P} = (\mathcal{R}, \mathcal{S}, \mathcal{A}, k)$ is* (ε, δ)-differentially private in the group shuffle model *if $S^k \circ \mathcal{R}^n(\vec{x}, w)$ is (ε, δ)-differentially private for every fixed $w \in \{0, 1\}^r$.*

Remark 1. *From the definition, we see that for $k = 1$ we obtain the definition of the shuffle model. On the other hand, if $k = n$, our model will coincide with the local privacy model (In this case, the shufflers do not do anything).*

3 Relation to LDP and Shuffle DP

In this section, we investigate the relation between our model, group shuffle, and other previous models – with a focus on the local and shuffle models. As discussed in Remark 1, those two models can be seen as the two extremes of the group shuffle model, for $k = n$ and $k = 1$, respectively. Thus, it is natural to ask about how the group shuffle model interpolates between them: that is, how the privacy/utility trade-off evolves, as a function of k. To start, we provide the following lemma, which establishes a necessary and sufficient condition for privacy in the group shuffle model.

Lemma 1. *A protocol $\mathcal{P} = (\mathcal{R}, \mathcal{S}, \mathcal{A}, k)$ is (ε, δ)-group shuffle private if, and only if, for all $\vec{x} \in \mathcal{X}^N$ and $i \in [k]$, $S_i(R(x_{(i-1)n'+1}, w), \ldots, R(x_{in'}, w))$ is (ε, δ)-differentially private for all $w \in \{0, 1\}^r$.*

In other words, $\mathcal{P}$ is (ε, δ)-group shuffle private if, and only if, the output of each of the k groups is individually (ε, δ)-shuffle private. We will extensively use this lemma in the remainder of this paper, as a crucial tool for our proofs.

Local model. In the local model (LDP), we assume the lowest level of confidence from the users towards the other parties, which as a consequence typically leads to much lower utility achievable. We next provide two theorems which relate the group shuffle and LDP models, and are extensions of previous, analogous theorems connecting shuffle and local models.

Theorem 2 (Extension of [6, Theorem 6.2])**.** *If a single-message protocol $\mathcal{P} = (\mathcal{R}, \mathcal{S}, \mathcal{A}, k)$ is (ε, δ)-group shuffle private, then $\mathcal{R}$ by itself satisfies $(\varepsilon + \ln \frac{n}{k}, \delta)$-differential privacy.*

The above theorem is an extension of [6, Theorem 6.2]. Its proof is straightforward and based on the equivalent definition of group shuffle DP given in Lemma 1. As the shuffle version from [6, Theorem 6.2], this lemma can be used to obtain lower bounds for group shuffle DP, from LDP lower bounds.

A powerful and appealing method for obtaining shuffle DP algorithms from LDP ones is the so-called "amplification by shuffling" [9, 1, 10]. In particular, the recent work [10] provides an optimal amplification lemma, showing how shuffling the output of LDP algorithms results in a shuffle private algorithms with much better (shuffle) privacy guarantees. We here extend their result, [10, Theorem 3.1], to the group shuffle model; for simplicity, and in view of our definition of group shuffle privacy, we only focus here on the case of non-adaptive protocols.

Theorem 3 (Extension of [10, Theorem 3.1] to Group Shuffle)**.** *Let $\mathcal{P} = (\mathcal{R}, \mathcal{A})$ be ε_L-LDP for n users, partitioned into k groups of equal size. Upon randomly permuting the concatenated outputs within each group, for every $\delta \in [0, 1]$ we obtain an (ε, δ)-group shuffle protocol with*

$$\varepsilon = \begin{cases} \mathcal{O}\left(\varepsilon_L \sqrt{\frac{k \log(1/\delta)}{n}}\right), & \varepsilon_L \leq 1 \\ \mathcal{O}\left(\sqrt{e^{\varepsilon_L} \frac{k \log(1/\delta)}{n}}\right), & \varepsilon_L > 1 \end{cases} \tag{3}$$

provided that $\varepsilon_L \leq \log \frac{n}{16k \log(2/\delta)}$.

Note that the distinction between this theorem and the original theorem is that we cannot uniformly permute all messages of users (equivalently, all inputs of users), but instead use a different and

independent permutation for each group. As a result, the amplification worsens by a factor roughly $\sqrt{k}$ (i.e., amplifying by roughly $\sqrt{k/n}$ instead of $1/\sqrt{n}$), which matches what one would expect for both extreme cases $k = 1$ and $k = n$.

Conversion to and from the shuffle model. Although we do not provide a generic, all-purpose conversion between group shuffle and shuffle models, we describe some relationships between the two in Appendix B.3.

4 The example of binary summation

In this section, we focus on the fundamental problem of *binary summation*, where each of the n users holds a bit $x_i \in \{0, 1\}$, and the goal is to privately estimate their sum $S(\vec{x}) = \sum_{i=1}^{n} x_i$. We analyze this particular task, extensively studied and well understood under DP, LDP, and shuffle DP, as a guiding example to understand the performance (accuracy) of algorithms in the group shuffle model.

Various protocols have been proposed for this problem in the shuffle DP literature, most of them yielding an unbiased estimator of the sum $S(\vec{x})$. We show how, leveraging this, one can derive a group shuffle protocol from a shuffle DP one. After that, we will show the optimality of our conversion, by lower bounding the MSE of any group shuffle protocol for binary summation.

Theorem 4. *Let $\mathcal{P}$ be (ε, δ)-shuffle private and its output be an unbiased estimate for the sum of n' binary inputs. Suppose that, for all inputs, the MSE of this protocol is $\mathcal{O}(g(\varepsilon, \delta, n'))$. Then, b concatenating the k independent instances of this protocol we obtain an (ε, δ)-group shuffle private protocol for $n \coloneqq kn'$ users whose MSE is $\mathcal{O}(k \cdot g(\varepsilon, \delta, n'))$.*

In particular, using the nearly optimal (pure) shuffle DP protocol of [11] for binary summation, we obtain the following:[1]

Corollary 5. *There exists an $(\varepsilon, 0)$-group shuffle private protocol $\mathcal{P}$ for binary summation with MSE $\mathcal{O}_\varepsilon(k)$, in which each user sends $\mathcal{O}_\varepsilon\big(\log \frac{n}{k}\big)$ single-bit messages.*

Note that the above conversion affects the MSE by a factor of k. It is natural to wonder if this loss is inherent in a conversion from shuffle to group shuffle DP. In our next theorem, we show that this is the case for binary summation, by lower bounding the MSE of any group shuffle private protocol.

Theorem 6. *Let $\mathcal{P}$ be any (ε, δ)-group shuffle private for binary summation. Then, the MSE of this protocol must be $\Omega\big(\frac{k}{\varepsilon^2}\big)$.*

Proof idea. The proof of this theorem relies on a conversion to the local model, for which we know that the MSE of any (ε, δ)-LDP protocol for binary summation is at least $\Omega\big(\frac{n}{\varepsilon^2}\big)$. Using this fact enables us to establish our lower bound via a reduction. □

5 Discussion and future work

We proposed a new privacy model, the *group shuffle model*, intermediate between the local and shuffle models and appropriate for situations the latter cannot be used, due, e.g., to implementation, policy, or security reasons. We investigated some of the relations between our model and existing ones in Section 3, and provided some conversion theorems. Moreover, we analyzed the type of privacy/utility tradeoffs one could expect in the group shuffle model, by focusing in Section 4 on the important example of binary summation, where we provided some upper and lower bounds. We believe investigating further the group shuffle model to be an appealing and well-motivated research direction. We list below some questions of interest.

- Can we design a new algorithm for binary summation which is specific to the group model, i.e., does not rely on a conversion from either the LDP or shuffle model?
- Is there a generic, more efficient conversion from a group shuffle private protocol to shuffle private one? How can we improve the proposed conversion?
- What is the optimal communication complexity achievable, and how does it interpolates between those of the local and shuffle models?
- What if we allow each group to have a different randomizer, instead of requiring all users to use the same? Can this lead to more efficient protocols?

[1] While [11] only proved an absolute error guarantee, one can check their algorithm achieves the stated MSE.

References

[1] Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. The privacy blanket of the shuffle model. In *CRYPTO (2)*, volume 11693 of *Lecture Notes in Computer Science*, pages 638–667. Springer, 2019.

[2] Amos Beimel, Kobbi Nissim, and Eran Omri. Distributed private data analysis: On simultaneously solving how and what, 2011.

[3] Andrea Bittau, Úlfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnés, and Bernhard Seefeld. Prochlo: Strong privacy for analytics in the crowd. In *SOSP*, pages 441–459. ACM, 2017.

[4] TH Hubert Chan, Elaine Shi, and Dawn Song. Optimal lower bound for differentially private multi-party aggregation. In *European Symposium on Algorithms*, pages 277–288. Springer, 2012.

[5] Albert Cheu. Differential privacy in the shuffle model: A survey of separations, 2021.

[6] Albert Cheu, Adam Smith, Jonathan Ullman, David Zeber, and Maxim Zhilyaev. Distributed differential privacy via shuffling. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 375–403. Springer, 2019.

[7] John C. Duchi, Michael I. Jordan, and Martin J. Wainwright. Local privacy and statistical minimax rates. In *FOCS*, pages 429–438. IEEE Computer Society, 2013.

[8] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography*, volume 3876 of *Lecture Notes in Comput. Sci.*, pages 265–284. Springer, Berlin, 2006. doi: 10.1007/11681878_14. URL `https://doi.org/10.1007/11681878_14`.

[9] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. Amplification by shuffling: From local to central differential privacy via anonymity. In *SODA*, pages 2468–2479. SIAM, 2019.

[10] Vitaly Feldman, Audra McMillan, and Kunal Talwar. Hiding among the clones: A simple and nearly optimal analysis of privacy amplification by shuffling, 2020.

[11] Badih Ghazi, Noah Golowich, Ravi Kumar, Pasin Manurangsi, Rasmus Pagh, and Ameya Velingker. Pure differentially private summation from anonymous messages. In *ITC*, volume 163 of *LIPIcs*, pages 15:1–15:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. URL `https://drops.dagstuhl.de/opus/volltexte/2020/12120/`.

[12] Shiva Prasad Kasiviswanathan, Homin K. Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? *SIAM J. Comput.*, 40(3):793–826, 2011. ISSN 0097-5397.

[13] Stanley L. Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965. doi: 10.1080/01621459.1965.10480775.

Appendices

A Proof of Lemma 1

Proof. We prove separately the two directions of the lemma. First, assume that the whole vector is (ε, δ)-differentially private, and fix $1 \leq i \leq k$. By post-processing, if we apply a function that maps the whole output message vector to only the i-th group's output, the result is still (ε, δ)-differentially private. This implies the output of i-th group is (ε, δ)-differentially private individually, completing one of the two directions.

Assume now that the output of each group is (ε, δ)-differentially private. Note that by the definition of our model, the outputs of the shufflers are independent. For ease of notation, we denote the input and output of the i-th group by X_i and $S_i(X_i)$ respectively. Let A be an arbitrary subset, and write it as the Cartesian product $A = A_1 \times \cdots \times A_k$, where A_i is a subset of the i-th shuffler's output space; we then have

$$\begin{aligned}\mathbb{P}[(S_1(X_1), S_2(X_2), \ldots, S_k(X_k)) \in A] &= \mathbb{P}[(S_1(X_1), S_2(X_2), \ldots, S_k(X_k)) \in A_1 \times \ldots \times A_k] \\ &= \mathbb{P}[S_1(X_1) \in A_1] \times \cdots \times \mathbb{P}[S_k(X_k) \in A_k]\,, \end{aligned} \tag{4}$$

where the last equality follows from the independence assumption of the S_is. Assume that X' is adjacent to X, meaning that both are the same except for a user's input in the j-th group. By definition of differential privacy for the j-th group we have

$$\mathbb{P}[S_j(X_j) \in A_j] \leq e^{\varepsilon} \cdot \mathbb{P}\big[S_j(X'_j) \in A_j\big] + \delta, \tag{5}$$

so that, multiplying both sides by $\prod_{\substack{1 \leq i \leq k \\ i \neq j}} \mathbb{P}[S_i(X_i) \in A_i]$ and using (4) yields that

$$\begin{aligned}\mathbb{P}[(S_1(X_1), S_2(X_2), \ldots, S_k(X_k)) \in A] &\leq \prod_{\substack{1 \leq i \leq k \\ i \neq j}} \mathbb{P}[S_i(X_i) \in A_i] \left(e^{\varepsilon} \cdot \mathbb{P}\big[S_j(X'_j) \in A_j\big] + \delta\right) \\ &= e^{\varepsilon} \cdot \mathbb{P}\big[(S_1(X_1), \ldots, S_j(X'_j), \ldots, S_k(X_k)) \in A_1 \times \ldots \times A_k\big] + \delta \prod_{\substack{1 \leq i \leq k \\ i \neq j}} \mathbb{P}[S_i(X_i) \in A_i] \\ &\leq e^{\varepsilon} \cdot \mathbb{P}[(S_1(X'_1), S_2(X'_2), \ldots, S_k(X'_k)) \in A] + \delta\,, \end{aligned} \tag{6}$$

where the last inequality uses that $\prod_{\substack{1 \leq i \leq k \\ i \neq j}} \mathbb{P}[S_i(X_i) \in A_i] \leq 1$. □

Remark 2. *Note that, in the above proof, we did not use the assumption that each group had the same number of members. Thus, the lemma generalizes to an extension of our model where each group can have a different number of users, instead of $n' = n/k$.*

B Relation to Other models

B.1 Proof of Theorem 2

To start, we state the general theorem of [6].

Theorem 7 ([6, Theorem 6.2])**.** *If a single-message protocol $\mathcal{P} = (\mathcal{R}, \mathcal{S}, \mathcal{A})$ is (ε, δ)-shuffle private, then $\mathcal{R}$ by itself satisfies $(\varepsilon + \ln n, \delta)$-differential privacy.*

Proof. From Lemma 1, we know that if $\mathcal{P} = (\mathcal{R}, \mathcal{S}, \mathcal{A}, k)$ is (ε, δ)-group shuffle private, then each output of groups' shufflers satisfies (ε, δ)-DP. As, in our model, each group consists of some users and a trusted shuffler, we can use Theorem 7 for each group individually. It follows that $\mathcal{R}$ should satisfies $(\varepsilon + \ln \frac{n}{k}, \delta)$-differential privacy. (Note that the original, as ours, focuses on the privacy, and thus the analyzer $\mathcal{A}$ itself is irrelevant to the result.) □

Remark 3. *As for Lemma 1, in the above proof, we did not require the assumption that groups had the same size. Specifically, each group had a different number of users, then we can obtain that $\mathcal{R}$ satisfies $(\varepsilon + \ln M, \delta)$-DP, where M is the minimum number of users in any of the k groups.*

B.2 Proof of Theorem 3

Before proving the theorem, we review the original theorem and its consequence.

Theorem 8 (Short version of [10, Theorem 3.1])**.** *Let $\mathcal{P} = (\mathcal{R}, \mathcal{A})$ be ε_L-LDP for n users. For every $\delta \in [0, 1]$ such that $\varepsilon_L \leq \log \frac{n}{16 \log(2/\delta)}$, by shuffling the outputs of the n users we obtain a protocol which is (ε, δ)-shuffle private, with*

$$\varepsilon \leq \log\left(1 + 8\frac{e^{\varepsilon_L} - 1}{e^{\varepsilon_L} + 1}\left(\frac{\sqrt{e^{\varepsilon_L}\log\frac{4}{\delta}}}{\sqrt{n}} + \frac{e^{\varepsilon_L}}{n}\right)\right). \tag{7}$$

In particular, as [10] stated, we have two regimes for ε with respect to ε_L,

$$\varepsilon = \begin{cases} \mathcal{O}\left(\varepsilon_L\sqrt{\frac{\log(1/\delta)}{n}}\right), & \varepsilon_L \leq 1 \\ \mathcal{O}\left(\sqrt{e^{\varepsilon_L}\frac{\log(1/\delta)}{n}}\right), & \varepsilon_L > 1. \end{cases} \tag{8}$$

Proof of Theorem 3. The proof is based on the converse side of Lemma 1. We use the amplification method of Theorem 8 for each group separately. Therefore, as all k groups have the same size, we can obtain (ε, δ)-differential privacy with

$$\varepsilon = \begin{cases} \mathcal{O}\left(\varepsilon_L\sqrt{\frac{k\log(1/\delta)}{n}}\right), & \varepsilon_L \leq 1 \\ \mathcal{O}\left(\sqrt{e^{\varepsilon_L}\frac{k\log(1/\delta)}{n}}\right), & \varepsilon_L > 1. \end{cases} \tag{9}$$

Now by using again Lemma 1, we know that if output of each group satisfies (ε, δ)-differential privacy, the whole concatenated vectors of all groups satisfies (ε, δ)-differential privacy; this concludes the proof. □

B.3 Relation to Shuffle models

We currently do not have a general conversion from shuffle to group shuffle model for all tasks. However, if we assume that the output of the analyzer is a specific statistic that can be calculated from many similar statistics (e.g, sum, frequency statistic (in some cases), etc.) then we can obtain a protocol in the group shuffle model from one in the shuffle model.

Specifically, we can simply concatenate the output of k shuffle protocols with $\frac{n}{k}$ users; the analyzer then first simulates separately the analyzers of each of the k groups, before aggregating those k outputs. Although this conversion is simple, it can in some cases lead to a tight or nearly tight bound in terms of accuracy for some tasks. We described this conversion in Section 4 for binary summation.

Note that the above is a conversion from k shuffle protocols on $\frac{n}{k}$ users to a group shuffle protocol on k groups of $\frac{n}{k}$ users. If we had shuffle protocols with more than $\frac{n}{k}$ users, then, by setting some inputs to a dummy value, e.g., 0 in the binary summation, the conversion would still go through.

Finally, we do not at the moment have any non-trivial conversion for the other direction, i.e., to derive a shuffle protocol from a group shuffle one; we see this as an interesting direction for future work.

C Completed proofs of binary sum problem

C.1 Proof of Theorem 4

Proof. The proof relies on a straightforward idea. Consider k instances of given $\mathcal{P}$. We assign each group of users to an instance. Now, we want to construct our group shuffle protocol: the randomizer of each user will be the same as the current protocol. We use a shuffler for each group. Clearly, by using Lemma 1, we have (ε, δ)-group shuffle privacy. In our analyzer, we first estimate the sum of each group using the given analyzer. After that, we simply obtain the output by summing up the k

estimations. For simplicity assume that the E_i is the Mean Squared Error of i-th group's estimation and E_T is our desired protocol MSE value. By using the fact that each estimation is unbiased, we get

$$E_T = E_1 + \ldots + E_k. \tag{10}$$

Furthermore, we assume that there exists a global function g such that for every input, E_i is $\mathcal{O}(g(\varepsilon, \delta, n'))$. Therefore, we have

$$E_T = k \cdot \mathcal{O}(g(\varepsilon, \delta, n')) = \mathcal{O}(k \cdot g(\varepsilon, \delta, n'))\,, \tag{11}$$

which completes the proof. □

C.2 Some notes on Corollary 5

Our starting point is [11, Algorithm 2]; which, once the expression for the output is corrected to $Y = \frac{n}{2} + \frac{1}{(1-p)}(\sum_{y \in R}(y - \frac{d}{2}))$, is indeed an unbiased estimate of the sum.[2] Now, we analyze the accuracy of the protocol. As we are interested in the MSE of protocol, we will bound the MSE, which is a (slightly) stronger statement than the expected absolute error bound they provide. We use their notation and show that the MSE is $\mathcal{O}\Big(\frac{\log \frac{1}{\varepsilon}}{\varepsilon^3}\Big)$.

Proof. Fix a dataset $\vec{x} = (x_1, \ldots, x_n)$. Assume that the output of protocol is Y, as given above; and denote the truncated discrete Laplace distribution by $\mathcal{D} \coloneqq \mathrm{DLap}_d(d/2, s)$, whose parameters are d and s. We have

$$\mathrm{MSE} = \mathbb{E}\left[(Y - \sum_{i=1}^{n} x_i)^2\right]. \tag{12}$$

Define $Z_i \coloneqq \frac{1}{1-p}(Y_i - \frac{d}{2}) + \frac{1}{2} - x_i$, where Y_i is equal to number of one received from the i-th user. It is clear that

$$Y - \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} Z_i \tag{13}$$

where the Z_i's are independent. Note that $\mathbb{E}_{Y' \sim \mathcal{D}}[Y']$ is equal to $\frac{d}{2}$, and so

$$\mathbb{E}[Z_i] = \frac{1}{2} - x_i + p\left(\frac{1}{1-p}(\mathbb{E}_{Y' \sim \mathcal{D}}[Y'] - \frac{d}{2})\right) + (1-p)\left(\frac{1}{1-p}\left(x_i - \frac{1}{2}\right)\right) = 0. \tag{14}$$

Hence,

$$\begin{aligned}
\mathrm{Var}[Z_i] = \mathbb{E}\big[Z_i^2\big] &= \frac{1}{(1-p)^2}\mathbb{E}\left[\left(Y_i - \frac{d}{2}\right)^2\right] - \left(\frac{1}{2} - x_i\right)^2 \\
&= \frac{1}{(1-p)}\left(x_i - \frac{1}{2}\right)^2 + \frac{p}{(1-p)^2}\,\mathrm{Var}_{Y' \sim \mathcal{D}}[Y'] - \left(\frac{1}{2} - x_i\right)^2 \\
&= \frac{p}{1-p}\left(\frac{1}{2} - x_i\right)^2 + \frac{p}{(1-p)^2}\,\mathrm{Var}_{Y' \sim \mathcal{D}}[Y'] \qquad (15) \\
&\leq \frac{p}{(1-p)^2}(\frac{1}{4}(1-p) + 2s^2) \qquad (16)
\end{aligned}$$

Where (16) obtained from the fact $\mathrm{Var}_{Y' \sim \mathcal{D}}[Y'] \leq 2s^2$. Finally, we have

$$\mathbb{E}\left[(Y - \sum_{i=1}^{n} x_i)^2\right] = \mathbb{E}\left[(\sum_{i=1}^{n} Z_i)^2\right] = \sum_{i=1}^{n} \mathbb{E}\big[Z_i^2\big] \leq \frac{np}{(1-p)^2}(\frac{1}{4}(1-p) + 2s^2).$$

We should set $s = \frac{10}{\varepsilon}$ and $p = \frac{100e^{100\varepsilon}\log(1/(1-e^{-0.1\varepsilon}))}{\lceil (n+1)/2 \rceil (1-e^{-0.1\varepsilon})}$, as per the privacy analysis of [11]. According to their assumption, we know that $\frac{1}{n^{2/3}} \leq \varepsilon < c_0$ holds where c_0 is a sufficient small constant. Now,

[2] We note that the typo in their expression for the output does not affect their privacy analysis.

while the above expression is a little unwieldy, one can check that $\frac{1}{1-p} = \mathcal{O}(1)$ (for sufficient large n). Hence,

$$\frac{np}{4(1-p)} + \frac{2nps^2}{(1-p)^2} = \frac{np}{4(1-p)} + \frac{200np}{\varepsilon^2(1-p)^2}$$
$$\leq \mathcal{O}\left(\frac{e^{100\varepsilon}\log(1/(1-e^{-0.1\varepsilon}))}{(1-e^{-0.1\varepsilon})}\right) + \mathcal{O}\left(\frac{e^{100\varepsilon}\log(1/(1-e^{-0.1\varepsilon}))}{\varepsilon^2(1-e^{-0.1\varepsilon})}\right).$$

We know that ε is upper bounded by c_0 and by using $\frac{1}{1-e^{-0.1\varepsilon}} \approx \frac{1}{0.1\varepsilon}$, we have

$$\mathcal{O}\left(\frac{e^{100\varepsilon}\log(1/(1-e^{-0.1\varepsilon}))}{(1-e^{-0.1\varepsilon})}\right) + \mathcal{O}\left(\frac{e^{100\varepsilon}\log(1/(1-e^{-0.1\varepsilon}))}{\varepsilon^2(1-e^{-0.1\varepsilon})}\right)$$
$$\leq \mathcal{O}\left(\frac{\log(\frac{1}{\varepsilon})}{\varepsilon}\right) + \mathcal{O}\left(\frac{\log(\frac{1}{\varepsilon})}{\varepsilon^3}\right) = \mathcal{O}\left(\frac{\log(\frac{1}{\varepsilon})}{\varepsilon^3}\right). \tag{17}$$

This completes the proof. Finally, note that by using the above upper bound along with Jensen's inequality, we get an expected absolute error of $\mathcal{O}\left(\frac{\sqrt{\log(\frac{1}{\varepsilon})}}{\varepsilon^{3/2}}\right)$, which retrieves the bound of [11]. □

C.3 Proof of Theorem 6

Before the proof, we recall the following theorem, which is [5, Theorem 9].

Theorem 9 ([4] and [2])**.** *Let $\mathcal{P}$ be (ε, δ)-local private protocol for binary summation with n users. If the additive error is up to α with constant probability, then we must have $\alpha = \Omega(\frac{\sqrt{n}}{\varepsilon})$.*

Further, it is known that the MSE of any locally private protocol for binary summation is $\Omega(\frac{n}{\varepsilon^2})$.

Proof. We prove this theorem by a reduction to the local model. Assume that we have k users. We randomly assign each user to a group. In each group, we set one input to the assigned user and set all other inputs to 0. We know that the output of each shuffler is (ε, δ)-differentially private with respect to the group's input. Therefore, in our construction, the outputs of groups are differentially private with respect to the assigned users. As a result, the whole protocol yields a local model protocol with k users. By invoking the MSE lower bound, stated above, we can thus bound the MSE of the protocol by $\Omega(\frac{k}{\varepsilon^2})$. This concludes the proof, as we established that the MSE of the original group shuffle protocol had to be $\Omega(\frac{k}{\varepsilon^2})$ in the worst case. □

Remark 4. *Note that in the proof, we only used the fact each group has at least an input. Therefore, even if the groups' sizes are not equal, the lower bound of $\Omega(\frac{k}{\varepsilon^2})$ still holds.*