Efficient Noise Generation Protocols for Differentially Private Multiparty Computation

Reo Eriguchi, Atsunori Ichikawa, Noboru Kunihiro, and Koji Nuida

Abstract—To bound information leakage in outputs of protocols, it is important to construct secure multiparty computation protocols which output differentially private values perturbed by the addition of noise. However, previous noise generation protocols have round and communication complexity growing with differential privacy budgets, or require parties to locally generate non-uniform noise, which makes it difficult to guarantee differential privacy against active adversaries. We propose three kinds of protocols for generating noise drawn from certain distributions providing differential privacy. The two of them generate noise from finite-range variants of the discrete Laplace distribution. For (ϵ, δ)-differential privacy, they only need constant numbers of rounds independent of ϵ, δ while the previous protocol needs the number of rounds depending on δ. The two protocols are incomparable as they make a trade-off between round and communication complexity. Our third protocol non-interactively generates shares of noise from the binomial distribution by predistributing keys for a pseudorandom function. It achieves communication complexity independent of ϵ or δ for the computational analogue of (ϵ, δ)-differential privacy while the previous protocols require communication complexity depending on ϵ. We also prove that our protocols can be extended so that they provide differential privacy in the active setting.

Index Terms—Differential privacy, secure multiparty computation, secret sharing.

1 Introduction

There is an increasing demand for providing statistical analysis of a large number of private data. A motivating example is performing surveys on customer information held by banks [1] or medical data stored by hospitals [2]. Secure multiparty computation (MPC) offers a solution, which enables parties to compute a function without revealing information on their data beyond an output. However, standard MPC protocols output an exact calculation result and cannot prevent an adversary from learning what follows from the result. In some applications, the exact value may contain some sensitive information on individuals. For example, exact statistics or machine learning models can be used to extract personal data [3], [4].

To deal with that problem, differentially private mechanisms [5], [6], [7] add noise drawn from an appropriate distribution to an exact calculation result and make the distributions of outputs for two "similar" inputs approximately the same. Since parties' inputs are sensitive, it is not appropriate to assume a trusted party who aggregates their data and applies a mechanism to them. We have to realize it in a distributed setting by emulating the trusted party.

- *Reo Eriguchi is with Graduate School of Information Science and Technology, The University of Tokyo, Bunkyo-ku, Tokyo 113–8654, Japan and also with National Institute of Advanced Industrial Science and Technology, Tokyo, Japan.*
  E-mail: reo-eriguchi@g.ecc.u-tokyo.ac.jp
- *Atsunori Ichikawa is with NTT Social Informatics Laboratories, Tokyo, Japan.*
  E-mail: atsunori.ichikawa.nf@hco.ntt.co.jp
- *Noboru Kunihiro is with Department of Computer Science, University of Tsukuba, Ibaraki, Japan.*
  E-mail: kunihiro@cs.tsukuba.ac.jp
- *Koji Nuida is with Institute of Mathematics for Industry, Kyushu University, Fukuoka, Japan and also with National Institute of Advanced Industrial Science and Technology.*
  E-mail: nuida@imi.kyushu-u.ac.jp

Generally, given an MPC protocol generating noise, we can emulate a mechanism based on the noise and obtain a protocol guaranteeing the same level of differential privacy [8], [9]. However, it is not straightforward to efficiently realize distributed noise generation since we have to correctly obtain noise drawn from a non-uniform distribution even if an adversary has access to and manipulates the local randomness of corrupted parties. Particularly, it cannot be solved by a simple protocol where a designated party samples and shares noise among the other parties, or where each party generates non-uniform noise r_i to perturb an outcome by the addition of $\sum_{i\in[n]} r_i$. Furthermore, many other differentially private protocols in the local model [10], [11], [12], [13] and in the shuffled model [14], [15], [16] also require parties to locally generate non-uniform noise, which makes it difficult in the presence of active adversaries to efficiently verify that corrupted parties indeed generate their randomness from the correct distribution.

Several suitable protocols for noise generation have been proposed in the literature [5], [9], [17]. Dwork et al. [5] devise secret-sharing based MPC protocols to generate shares of noise drawn from the discrete Laplace distribution and the binomial distribution of parameter $1/2$. However, their protocol for the discrete Laplace distribution requires the number of rounds proportional to approximately $\log\log\delta^{-1}$ to achieve (ϵ, δ)-differential privacy. Their protocols for the binomial distribution have to securely generate almost the same number of uniform random bits as the size of the support of the distribution. Then, the communication complexity grows linearly in ϵ^{-2} and exponentially in the length of the fractional part when a fixed-point data type is dealt with. The protocols [9], [17] assume arithmetic operations for real numbers with infinite precision and do not rigorously analyze achievable levels of differential privacy when they are implemented under finite-precision semantics.

In summary, there is no constant-round protocol for generating discrete Laplace noise with theoretical guarantees of differential privacy, or no protocol achieving communication complexity independent of privacy budgets ϵ, δ for binomial noise.

1.1 Our Results

We propose MPC protocols achieving constant round complexity for the discrete Laplace distribution and achieving communication complexity independent of ϵ, δ for the binomial distribution. We rigorously analyze achievable levels of differential privacy of them. They can also be extended so that they provide differential privacy against active adversaries.

1.1.1 Abstraction of an Output Perturbation Framework for MPC

We first abstract a framework implicitly used in the previous protocols [5], [9], [17]. Specifically, they let parties agree on a uniformly random element $\boldsymbol{s} = (s_i)_{i \in [n]}$ of a subset $U \subseteq S^n$ for some set S, e.g., the set of all possible shares of 0 and 1. They then securely compute a deterministic function $h(\boldsymbol{s})$, which we term a noise generator function. Assume that the distribution of $h(\boldsymbol{s})$ provides (ϵ, δ)-differential privacy even conditioned on $(s_i)_{i \in T}$ being fixed where $T \subseteq [n]$. Then, we can generally construct an (ϵ, δ)-differentially private protocol against corruption of T from protocols securely sampling $\boldsymbol{s} \in U$ and computing $h(\boldsymbol{s})$. An important advantage of this framework is that distributed noise generation is reduced to secure computation of *deterministic* functions and generation of *uniformly random* elements. We can cut down communication cost by devising an instantiation of h and achieve differential privacy against active adversaries by using the known actively secure protocols [18], [19].

1.1.2 Two Novel Noise Generation Protocols for the Discrete Laplace Distribution

We instantiate h with two noise generator functions whose outputs on a random input follow finite-range variants of the discrete Laplace distribution. As a result, we obtain constant-round protocols improving the round complexity of [5]. For example, while the previous protocol [5] needs 25 rounds in practical parameter settings $\epsilon = 0.5$ and $\delta = 2^{-60}$ [20], our first protocol only requires 19 rounds and our second one 14 rounds for any choice of ϵ, δ. Furthermore, the probability that protocols fail to generate noise is negligible in our first protocol and 0 in our second one while it is non-negligible in [5]. It means that a single protocol execution would be sufficient and our protocols can achieve better utility than [5]. Although our protocols have higher communication complexity than [5], it does not weaken our advantages in round complexity. Actually, our protocols reduce the estimated total running time of [5] by around $20\,\%$ in the above parameter setting. As for a comparison between our protocols, the first one has lower communication cost but needs more rounds of interaction than the second one. As a result, the first one is faster for small ϵ while the second one is faster for large ϵ. The second one has another advantage of high utility coming from the failure probability being zero and hence is even applicable to ϵ such that the first one is not. A more detailed comparison is given in Section 6.1.

1.1.3 A Novel Noise Generation Protocol for the Binomial Distribution

We also instantiate h with a function whose output on a random input follows the binomial distribution of size N and parameter $1/2$. Based on it, we propose a protocol that enables parties to locally compute shares of binomial noise by predistributing keys for a pseudorandom function. Technically, it is a modification of pseudorandom secret sharing [21]. Our protocol reduces the previous communication complexity linear in N [5] at the cost of predistributing keys. Furthermore, once the keys are distributed, parties can non-interactively generate shares of polynomially many binomial samples, which amortizes the communication cost in the setup. As a drawback, our protocol only satisfies the computational analogue of differential privacy [22]. Moreover, the error bound of our protocol is $O(n^{t/2})$ times larger than that of [5], where t is a corruption threshold. Nevertheless, our protocol works well in the client-server model, in which n corresponds to the number of servers and is typically small, e.g., $n = 3$. We compare the estimated total running time of ours with [5], and figure out that it is up to 23 times faster when generating 100 binomial noises for $\epsilon = 0.5$ and $\delta = 2^{-60}$. A more detailed comparison is given in Section 6.2.

1.1.4 Extension to Active Security

To the best of our knowledge, we are first to formalize unconditional security of MPC achieving differential privacy against active adversaries in the ideal-real-world paradigm. Although the authors of [17], [23] extend their protocols to the active setting, their security depends on cryptographic primitives such as computational verifiable secret sharing and zero-knowledge proof, and they do not discuss unconditional security. We prove that our general framework provides unconditionally secure protocols against active adversaries given actively secure protocols computing deterministic functions and generating uniformly random elements. In particular, our above protocols can be extended to the active setting by replacing the underlying MPC primitives accordingly.

For concrete efficiency, we estimate the running times for our actively secure protocols to generate samples from the discrete Laplace and binomial distributions. Calculating additional communication bits and rounds of interaction, we figure out that our actively secure protocols are about 4–16 times slower than the passively secure counterparts. Based on the experimental results [24], we also see that our protocols can generate noises in the presence of active adversaries for a reasonable time even if local computation time is taken into account. A more detailed performance evaluation is given in Section 7.

1.2 Related Work

The authors of [9], [17], [23] construct protocols for generating noise drawn from the Gaussian and Laplace distributions by making black-box use of MPC protocols for operations over real numbers. However, since secure computation over real numbers is more costly than on integers (e.g., more than 140 rounds are necessary for computing logarithms in the floating-point representation [25]), our protocols outperform the protocols [9], [17] in efficiency when an integer data type is dealt with. Furthermore, they lack a rigorous analysis of the impact of the finite-precision implementations on differential privacy, which is undesirable since differentially private mechanisms are vulnerable to the inexact computations [26], [27].

There are many differentially private mechanisms in the local model [10], [11], [12], [13], in which every party locally randomizes his private input and sends it to a designated party, who then computes a function on the noisy data. However, as mentioned above, local-model mechanisms require parties to locally generate non-uniform noise and it is difficult to efficiently verify the correctness of local randomness of corrupted parties in the presence of active adversaries. In addition, since a careful analysis of accumulated noise is needed, mechanisms are proposed only for a limited class of functions such as aggregate-sum queries $\sum_{i\in[n]} f_i(x_i)$. Other mechanisms in the shuffled model [14], [15], [16] also involve non-uniform noise generation and are applicable only for simple functions. We refer the reader to [28] for a survey.

The authors of [29] propose a method to generate many biased bits improving the efficiency of [5]. However, the method is based on oblivious data structures and is not directly applicable to secret-sharing based MPC protocols.

1.3 Publication Note

The preliminary version appeared in the proceedings of Financial Cryptography and Data Security 2021 [30]. The current version provides a novel protocol for the discrete Laplace distribution improving the communication complexity of [30]. In addition, this paper proves that our protocols can be extended to the active setting.

2 Preliminaries

2.1 Notations

For $n \in \mathbb{N}$, $[n]$ denotes $\{z \in \mathbb{Z} : 1 \leq z \leq n\}$ and $[0..n)$ denotes $\{z \in \mathbb{Z} : 0 \leq z \leq n-1\}$. We denote by e^x or $\exp(x)$ the exponential function of $x \in \mathbb{R}$. We assume that q is a sufficiently large odd prime and identify the prime field $\mathbb{Z}_q$ of size q with $\{z \in \mathbb{Z} : -q/2 < z < q/2\}$. A function $f : \mathbb{N} \to \mathbb{R}$ is negligible in λ and denoted by $\mathsf{negl}(\lambda)$ if for any $c > 0$, there exists $N \in \mathbb{N}$ such that $0 \leq f(\lambda) < 1/\lambda^c$ for any $\lambda \geq N$. For $T \subseteq [n]$ and a vector $\boldsymbol{x} \in S^n$, we denote by $\boldsymbol{x}_T \in S^{|T|}$ the sub-vector obtained by restricting the indices to T.

Let X and Y be two random variables with range U. We define the statistical distance $\mathsf{SD}(X, Y)$ between X and Y as $\mathsf{SD}(X, Y) = (1/2)\sum_{u\in U} |\Pr[X = u] - \Pr[Y = u]|$. It holds that $\mathsf{SD}((X_1, X_2), (Y_1, Y_2)) \leq \mathsf{SD}(X_1, Y_1) + \mathsf{SD}(X_2, Y_2)$ for random variables X_1, X_2, Y_1, Y_2 and that $\mathsf{SD}(F(X), F(Y)) \leq \mathsf{SD}(X, Y)$ for any randomized function F. We denote by $X \sim \mathcal{D}$ if X is distributed according to a probability distribution $\mathcal{D}$. We also write $s \sim \mathcal{D}$ if s is a value sampled from $\mathcal{D}$. We denote by $\mathsf{Uni}(S)$ the uniform distribution over a finite set S.

2.2 Secure Multiparty Computation

Assume that there are n parties holding their private inputs x_i, $i \in [n]$ from a finite set D. Let g be a deterministic function to compute on their inputs. Let Π be a protocol. For a subset $T \subseteq [n]$, we define $\mathsf{View}_T^{\Pi}(\boldsymbol{x})$ as the joint view of the parties in T during the execution of Π with inputs $\boldsymbol{x}$. We say that Π is an MPC protocol t-securely computing g if (1) for any subset $T \subseteq [n]$ of size t and any pair of inputs $\boldsymbol{x} = (x_i)_{i\in[n]}$ and $\boldsymbol{y} = (y_i)_{i\in[n]}$, the distributions of $\mathsf{View}_T^{\Pi}(\boldsymbol{x})$ and $\mathsf{View}_T^{\Pi}(\boldsymbol{y})$ are identical as long as $\boldsymbol{x}_T = \boldsymbol{y}_T$ and $g(\boldsymbol{x}) = g(\boldsymbol{y})$ and (2) for every $\boldsymbol{x}$ and $i \in [n]$, $g(\boldsymbol{x})$ is included in $\mathsf{View}_i^{\Pi}(\boldsymbol{x})$. See [31] for a more general case of n-input/n-output randomized functionalities in the presence of an active adversary.

2.2.1 *Secret Sharing*

Given a secret $a \in \mathbb{Z}_q$, the (t, n)-Shamir secret sharing scheme [32] generates a random polynomial p of degree at most t such that $p(0) = a$ and outputs $[\![a]\!]_i = p(i)$ as the i-th share. We simply write $[\![a]\!]$ if the index i is clear from the context. If $t < n/2$, there exists a protocol MULT t-securely computes $[\![ab]\!]$ from $[\![a]\!]$ and $[\![b]\!]$ [33]. If $t < n/3$, it is possible to t-securely realize it even in the presence of active adversaries. We measure the communication complexity of an MPC protocol by the number of invocations of MULT and its round complexity by the number of sequential invocations of MULT.

2.2.2 *Pseudorandom Secret Sharing*

Pseudorandom secret sharing [21] allows parties to non-interactively generate shares of a pseudorandom number by predistributing keys for a pseudorandom function. Technically, a pseudorandom function [34] with length parameters $s, \ell : \mathbb{N} \to \mathbb{N}$ is a collection of functions $\{\psi_r : \{0,1\}^{s(\lambda)} \to \{0,1\}^{\ell(\lambda)}\}_{r\in\{0,1\}^*}$, where $\{0,1\}^*$ denotes the set of all the bit strings of arbitrary length and λ is the bit length of r, such that (efficient evaluation) $\psi_r(a)$ can be computed in polynomial time from $r \in \{0,1\}^\lambda$ and $a \in \{0,1\}^{s(\lambda)}$ and (pseudorandomness) for every probabilistic polynomial-time (PPT) oracle machine M which has access to outputs of a function on inputs of its choice, it holds that $|\Pr[\mathsf{M}^{\psi_{U_\lambda}}(1^\lambda) = 1] - \Pr[\mathsf{M}^{F_\lambda}(1^\lambda) = 1]| = \mathsf{negl}(\lambda)$, where $U_\lambda \sim \mathsf{Uni}(\{0,1\}^\lambda)$ and F_λ is a uniformly selected map from $\{0,1\}^{s(\lambda)}$ to $\{0,1\}^{\ell(\lambda)}$. There is a provably secure pseudorandom function $\{\psi_r : \{0,1\}^s \to \{0,1\}^\ell\}_{r\in\{0,1\}^\lambda}$ with $\lambda = 161 \times 160$ and $s = \ell = 160$ [35]. One can also use the AES encryption with $\lambda = 128$ and $s = \ell = 128$.

Let $\mathcal{A} = \{A \subseteq [n] : |A| = n - t\}$ and $\mathcal{A}_i = \{A \in \mathcal{A} : i \in A\}$ for $i \in [n]$. For $A \in \mathcal{A}$, let $f_A \in \mathbb{Z}_q[X]$ be the unique polynomial such that $f_A(0) = 1$, $f_A(i) = 0$ for any $i \in [n] \setminus A$, and $\deg f_A \leq t$. Assume that for each $A \in \mathcal{A}$, parties in A receive a key $r_A \sim \mathsf{Uni}(\{0,1\}^\lambda)$. The i-th party locally computes $v_i = \sum_{A\in\mathcal{A}_i} \psi_{r_A}(a) f_A(i)$ from his keys $(r_A)_{A\in\mathcal{A}_i}$ and a public input $a \in \{0,1\}^{s(\lambda)}$ by embedding $\{0,1\}^{\ell(\lambda)}$ into $\mathbb{Z}_q$. It can be verified that $(v_i)_{i\in[n]}$ are consistent shares of the (t, n)-Shamir secret sharing scheme for a pseudorandom number $\sum_{A\in\mathcal{A}} \psi_{r_A}(a) \in \mathbb{Z}_q$. Note that the setup assumption can be removed by using any protocol for generating random shares of the replicated secret sharing scheme (e.g. [36]) since $(r_A)_{A\in\mathcal{A}_i}$ is the i-th share of the replicated secret sharing scheme for a secret $\bigoplus_{A \in \mathcal{A}} r_A \in \{0,1\}^\lambda$, where $\bigoplus$ is the bit-wise xor operation.

2.2.3 Primitives

We introduce constant-round MPC protocols for specific functionalities. A protocol $\textsc{Pre}_\vee^\ell$ securely computes ℓ shares $[\![\vee_{k=1}^j a_k]\!]$, $j \in [\ell]$ from $[\![a_k]\!]$, $k \in [\ell]$ if $q > 2\ell$. It needs 7 rounds (including 2 rounds for random value generation) and its communication complexity is 17ℓ invocations [37]. A protocol Σ_{Bit} generates shares for a uniform random bit with 2 rounds and 2 invocations [38]. Nishide and Ohta [37] propose a protocol for the interval test with 13 rounds (including 2 rounds for random value generation) and $110 \log q + 1$ invocations. Specifically, it securely computes $[\![c]\!]$ from $[\![x]\!]$ and public constants $a, b \in \mathbb{Z}_q$, where $c = 1$ if $a \le x \le b$ and $c = 0$ otherwise. They also propose a protocol for the equality test computing $[\![c]\!]$ from $[\![x]\!]$ and $[\![y]\!]$, where $c = 1$ if $x = y$ and $c = 0$ otherwise. Its round complexity is 8 rounds (including 2 rounds for random value generation) and communication complexity is $81 \log q$ invocations. Catrina and Hoogh [39] propose a protocol IP^ℓ with 1 round and 1 invocation computing $[\![\sum_{k=1}^\ell a_k b_k]\!]$ from $[\![a_k]\!]$ and $[\![b_k]\!]$ for $k \in [\ell]$.

2.3 Differential Privacy

Two random variables X, Y are said to be (ϵ, δ)-DP close if for every distinguisher D, it holds that

$$\Pr[\mathsf{D}(X) = 1] \le e^\epsilon \Pr[\mathsf{D}(Y) = 1] + \delta.$$

Analogously, we say that they are computationally (ϵ, δ)-DP close [22] if for every PPT distinguisher D, it holds that

$$\Pr[\mathsf{D}(X) = 1] \le e^\epsilon \Pr[\mathsf{D}(Y) = 1] + \delta + \mathsf{negl}(\lambda),$$

where λ is a security parameter. Two vectors $\boldsymbol{x} = (x_1, \ldots, x_n)$, $\boldsymbol{y} = (y_1, \ldots, y_n) \in D^n$ are called T-neighboring if there is exactly one index $i \in [n] \setminus T$ such that $x_i \ne y_i$, and simply called neighboring if they are $\emptyset$-neighboring. We say that a randomized functionality $\mathcal{M}$ with domain D^n is (resp. computationally) (ϵ, δ)-differentially private if $\mathcal{M}(\boldsymbol{x})$ and $\mathcal{M}(\boldsymbol{y})$ are (resp. computationally) (ϵ, δ)-DP close for all neighboring vectors $\boldsymbol{x}, \boldsymbol{y} \in D^n$. When the output domain of $\mathcal{M}$ is included in $\mathbb{R}$, we say that $\mathcal{M}$ satisfies (α, β)-utility for a function $g : D^n \to \mathbb{R}$ if for every input $\boldsymbol{x} \in D^n$, it holds that $\Pr[|\mathcal{M}(\boldsymbol{x}) - g(\boldsymbol{x})| \le \alpha] \ge 1 - \beta$. We only consider passive adversaries who see internal states of corrupted parties but do not deviate from protocols until Section 7. For now, we suppose the following definition of MPC protocols satisfying differential privacy against passive adversaries.

Definition 1. An MPC protocol Π is called a $(t; \epsilon, \delta)$-differentially private protocol for computing g with (α, β)-utility if the following holds:

- t-Privacy: for every subset T of size at most t and for any pair of inputs $\boldsymbol{x}$ and $\boldsymbol{y}$ with $\boldsymbol{x}_T = \boldsymbol{y}_T$ and $g(\boldsymbol{x}) = g(\boldsymbol{y})$, the distributions of $\mathsf{View}_T^\Pi(\boldsymbol{x})$ and $\mathsf{View}_T^\Pi(\boldsymbol{y})$ are identical.
- (ϵ, δ)-Differential privacy: for every subset T of size at most t and for any pair of T-neighboring inputs $\boldsymbol{x}$ and $\boldsymbol{y}$, the distributions of $\mathsf{View}_T^\Pi(\boldsymbol{x})$ and $\mathsf{View}_T^\Pi(\boldsymbol{y})$ are (ϵ, δ)-DP close.
- (α, β)-Utility: for every input $\boldsymbol{x}$, it holds that $\Pr[|\Pi(\boldsymbol{x}) - g(\boldsymbol{x})| \le \alpha] \ge 1 - \beta$, where $\Pi(\boldsymbol{x})$ is an output of Π on input $\boldsymbol{x}$.

We can define a computational analogue of $(t; \epsilon, \delta)$-differentially private protocols by considering (ϵ, δ)-DP closeness in the computational setting. Note that we do not relax t-privacy in the computational setting.

In the above definition, we require that differential privacy holds for pairs of T-neighboring inputs for a set T of corrupted parties, instead of $\emptyset$-neighboring ones. This is because what we need to guarantee is that an adversary cannot tell if at most one honest party $i \notin T$ changes his input in the sense of differential privacy. For neighboring inputs differing on an entry whose index is in T, the adversary can trivially break differential privacy by viewing the corrupted parties' inputs. We note that this definition is adopted in the literature [8], [9].

2.4 Additive Noise Mechanisms

A typical technique to achieve differential privacy is adding controlled noise to an outcome. Let $g : D^n \to \mathbb{R}$. We define the sensitivity Δ of g as $\Delta = \max\{|g(\boldsymbol{x}) - g(\boldsymbol{y})| : \boldsymbol{x}$ and $\boldsymbol{y}$ are neighboring$\}$ and the range R of g as $R = \max_{\boldsymbol{x} \in D^n} |g(\boldsymbol{x})|$.

For $0 < p < 1$, the discrete Laplace distribution $\mathsf{DL}(p)$ is the distribution of L defined as $\Pr[L = k] = p^{|k|}(1-p)/(1+p)$ for $k \in \mathbb{Z}$ [40]. The functionality $g(\cdot) + L$ satisfies $(\epsilon, 0)$-differential privacy and (α, β)-utility if $p = \exp(-\epsilon/\Delta)$ and $\alpha = (\Delta/\epsilon)\ln(1/\beta)$ [41]. Since we focus on integers on a finite interval, we consider finite-range variants of $\mathsf{DL}(p)$, which are also shown to provide differential privacy in Section 4.

The binomial distribution is also used in the literature [7]. For $\ell, M \in \mathbb{N}$, let $Y \sim \mathsf{Bin}(N, 1/2)$, i.e., $\Pr[Y = k] = \binom{N}{k} 2^{-N}$ for $k = 0, 1, \ldots, N$ and define $\mathsf{NBin}(N, M)$ as the distribution of $(1/M)\,(Y - N/2)$. Suppose that N, M satisfy $N/4 \ge \max\{23 \log(10/\delta), 2\Delta M\}$ for some $\delta > 0$. For $Z \sim \mathsf{NBin}(N, M)$, the functionality $\mathcal{M}_g^{N,M}(\cdot) = g(\cdot) + Z$ is (ϵ, δ)-differentially private if

$$\epsilon \ge \epsilon(\delta, N, M, \Delta) := \Delta \left(c_1(\delta) \frac{M}{\sqrt{N}} + c_2(\delta) \frac{M}{N} \right), \tag{1}$$

where $c_1(\delta) = O(\sqrt{\log \delta^{-1}})$ and $c_2(\delta) = O((\log \delta^{-1})^2)$. For any input $\boldsymbol{x} \in D^n$, Hoeffding's inequality implies that $\mathcal{M}_g^{N,M}$ satisfies (α, β)-utility for g if $\beta = 2\exp(-2M^2\alpha^2/N)$, i.e., $\alpha = \sqrt{(N/2M^2)\ln(2/\beta)}$.

3 Abstraction of an Output Perturbation Framework for MPC

We abstract out the framework implicitly used in the previous protocols for generating shares of noise drawn from non-uniform distributions [5], [9], [17]. They first get parties agree on an element uniformly selected from a certain set and then securely compute a deterministic function h on the element. Technically, let S be a finite set and U be a subset of S^n. For $T \subseteq [n]$, we define $U_T = \{\boldsymbol{a} \in S^{|T|} : \boldsymbol{a} = \boldsymbol{u}_T$ for some $\boldsymbol{u} \in U\}$.

Assumption. The parties receive correlated randomness $\boldsymbol{s} \sim \mathsf{Uni}(U)$.
Input. $\boldsymbol{x} \in D^n$.
Output. $z = g(\boldsymbol{x}) + h(\boldsymbol{s}) \in \mathbb{Z}_q$.
Protocol.

1) $[\![g(\boldsymbol{x})]\!] = \Pi_g(\boldsymbol{x})$.
2) $[\![h(\boldsymbol{s})]\!] = \Pi_h(\boldsymbol{s})$.
3) $[\![z]\!] = [\![g(\boldsymbol{x})]\!] + [\![h(\boldsymbol{s})]\!]$.
4) Reconstruct and output z.

Fig. 1: A $(t;\epsilon,\delta)$-differentially private protocol for computing g

Proposition 1. Let $g : D^n \to \mathbb{Z}_q$ and $h : U \to \mathbb{Z}_q$ be deterministic functions. Let Π_g (resp. Π_h) be a protocol which takes $\boldsymbol{x} \in D^n$ (resp. $\boldsymbol{s} \in U$) as input and t-securely computes $([\![g(\boldsymbol{x})]\!]_i)_{i\in[n]}$ (resp. $([\![h(\boldsymbol{s})]\!]_i)_{i\in[n]}$). Assume that, for any subset T of size t, any pair of T-neighboring vectors $\boldsymbol{x}, \boldsymbol{y} \in D^n$, and any $\boldsymbol{a} \in U_T$, two distributions $(\boldsymbol{x}_T, \boldsymbol{s}_T, g(\boldsymbol{x})+h(\boldsymbol{s})), (\boldsymbol{y}_T, \boldsymbol{s}_T, g(\boldsymbol{y})+h(\boldsymbol{s}))$ conditioned on $\boldsymbol{s}_T = \boldsymbol{a}$ are (ϵ,δ)-DP close over the randomness of $\boldsymbol{s} \sim \mathsf{Uni}(U)$. Furthermore, assume that $\Pr_{\boldsymbol{s}\sim\mathsf{Uni}(U)}[|h(\boldsymbol{s})| \le \alpha] \ge 1-\beta$ for α and β. Then, the protocol Π described in Fig. 1 (assuming certain correlated randomness) is a $(t;\epsilon,\delta)$-differentially private protocol for computing g with (α,β)-utility.

We call h in Proposition 1 a noise generator function. We emphasize that the level of differential privacy and utility that the resultant protocol achieves only depends on the property of h. Therefore, a single noise generator function h can give the same level of differential privacy and utility to MPC protocols for many different functions g.

Proof of Proposition 1: Let $T \subseteq [n]$ with $|T| \le t$. For $\boldsymbol{x} \in D^n$ and *fixed* $\boldsymbol{s} \in U$, let $v_T^\Pi(\boldsymbol{x};\boldsymbol{s})$ denote the joint view of T when Π is executed on $\boldsymbol{x}$ and $\boldsymbol{s}$. Note that the distribution of $\mathsf{View}_T^\Pi(\boldsymbol{x})$ is the same as that of $v_T^\Pi(\boldsymbol{x};\boldsymbol{s})$ induced by $\boldsymbol{s} \sim \mathsf{Uni}(U)$. From the security of Π_g and Π_h, we have a simulator Sim_T such that the distribution of $v_T^\Pi(\boldsymbol{x};\boldsymbol{s})$ is the same as that of $\mathsf{Sim}_T(\boldsymbol{x}_T, \boldsymbol{s}_T, g(\boldsymbol{x})+h(\boldsymbol{s}))$ for fixed $\boldsymbol{x}$ and $\boldsymbol{s}$.

As for t-privacy, let $\boldsymbol{x}, \boldsymbol{y}$ be two inputs such that $\boldsymbol{x}_T = \boldsymbol{y}_T$ and $g(\boldsymbol{x}) = g(\boldsymbol{y})$. Then, it holds that for any set of transcripts $\mathcal{O}$,

$$\begin{aligned}
&\Pr\left[\mathsf{View}_T^\Pi(\boldsymbol{x}) \in \mathcal{O}\right] \\
&= \Pr_{\boldsymbol{s}\sim\mathsf{Uni}(U)}\left[v_T^\Pi(\boldsymbol{x};\boldsymbol{s}) \in \mathcal{O}\right] \\
&= \sum_{\boldsymbol{u}\in U} \Pr[\boldsymbol{s}=\boldsymbol{u}] \cdot \Pr\left[v_T^\Pi(\boldsymbol{x};\boldsymbol{u}) \in \mathcal{O}\right] \\
&= \sum_{\boldsymbol{u}\in U} \Pr[\boldsymbol{s}=\boldsymbol{u}] \cdot \Pr[\mathsf{Sim}_T(\boldsymbol{x}_T, \boldsymbol{u}_T, g(\boldsymbol{x})+h(\boldsymbol{u})) \in \mathcal{O}] \\
&= \sum_{\boldsymbol{u}\in U} \Pr[\boldsymbol{s}=\boldsymbol{u}] \cdot \Pr[\mathsf{Sim}_T(\boldsymbol{y}_T, \boldsymbol{u}_T, g(\boldsymbol{y})+h(\boldsymbol{u})) \in \mathcal{O}] \\
&= \Pr\left[\mathsf{View}_T^\Pi(\boldsymbol{y}) \in \mathcal{O}\right].
\end{aligned}$$

As for (ϵ,δ)-differential privacy, let $\boldsymbol{x}, \boldsymbol{y}$ be T-neighboring inputs. From the assumption, we have that for any set of outcomes $\mathcal{O}'$ and any $\boldsymbol{a} \in U_T$,

$$\begin{aligned}
&\Pr[(\boldsymbol{x}_T, \boldsymbol{s}_T, g(\boldsymbol{x})+h(\boldsymbol{s})) \in \mathcal{O}' \mid \boldsymbol{s}_T = \boldsymbol{a}] \\
&\le e^\epsilon \Pr[(\boldsymbol{y}_T, \boldsymbol{s}_T, g(\boldsymbol{y})+h(\boldsymbol{s})) \in \mathcal{O}' \mid \boldsymbol{s}_T = \boldsymbol{a}] + \delta.
\end{aligned}$$

Since differential privacy is immune to post-processing, it also holds that for any set of transcripts $\mathcal{O}$ and any $\boldsymbol{a} \in U_T$,

$$\begin{aligned}
&\Pr[\mathsf{Sim}_T(\boldsymbol{x}_T, \boldsymbol{s}_T, g(\boldsymbol{x})+h(\boldsymbol{s})) \in \mathcal{O} \mid \boldsymbol{s}_T = \boldsymbol{a}] \\
&\le e^\epsilon \Pr[\mathsf{Sim}_T(\boldsymbol{y}_T, \boldsymbol{s}_T, g(\boldsymbol{y})+h(\boldsymbol{s})) \in \mathcal{O} \mid \boldsymbol{s}_T = \boldsymbol{a}] + \delta.
\end{aligned}$$

Since $v_T^\Pi(\boldsymbol{x};\boldsymbol{s})$ and $\mathsf{Sim}_T(\boldsymbol{x}_T, \boldsymbol{s}_T, g(\boldsymbol{x})+h(\boldsymbol{s}))$ are identically distributed for fixed $\boldsymbol{x}$ and $\boldsymbol{s}$, we obtain that for any set of transcripts $\mathcal{O}$ and any $\boldsymbol{a} \in U_T$,

$$\begin{aligned}
&\Pr\left[v_T^\Pi(\boldsymbol{x};\boldsymbol{s}) \in \mathcal{O} \,\middle|\, \boldsymbol{s}_T = \boldsymbol{a}\right] \\
&\le e^\epsilon \Pr\left[v_T^\Pi(\boldsymbol{y};\boldsymbol{s}) \in \mathcal{O} \,\middle|\, \boldsymbol{s}_T = \boldsymbol{a}\right] + \delta.
\end{aligned}$$

Thus, taking the probability-weighted sum over all $\boldsymbol{a} \in U_T$ gives $\Pr\left[\mathsf{View}_T^\Pi(\boldsymbol{x}) \in \mathcal{O}\right] \le e^\epsilon \Pr\left[\mathsf{View}_T^\Pi(\boldsymbol{y}) \in \mathcal{O}\right] + \delta$ for any set of transcripts $\mathcal{O}$.

Finally, (α,β)-utility easily follows. □

It can be seen that Π is a computationally $(t;\epsilon,\delta)$-differentially private protocol if h only provides (ϵ,δ)-differential privacy in the computational setting.

If generating a random input to h is equivalent to obtaining shares for uniform random values or bits, we can construct differentially private protocols without the setup assumption by running protocols for generating random shares. This is actually the case in all of the previous protocols [5], [9], [17] and our instantiations given later.

As a corollary of Proposition 1, if the input domain of h is the set of all the possible shares of m secret bits and the output of h essentially depends on the m bits, then it becomes easier to find out the level of differential privacy that h can provide.

Corollary 1. Let g and Π_g be as in Proposition 1. Let $h_0 : \{0,1\}^m \to \mathbb{Z}_q$ be a deterministic function and Π_0 be a protocol which takes shares of m bits $b = (b_1, \ldots, b_m)$ as input and t-securely computes $([\![h_0(b)]\!]_i)_{i\in[n]}$. Assume that, for any subset T of size t and any pair of T-neighboring vectors $\boldsymbol{x}, \boldsymbol{y} \in D^n$, two distributions $(\boldsymbol{x}_T, g(\boldsymbol{x})+h_0(b)), (\boldsymbol{y}_T, g(\boldsymbol{y})+h_0(b))$ are (ϵ,δ)-DP close over the randomness of $b \sim \mathsf{Uni}(\{0,1\}^m)$. Furthermore, assume that $\Pr_{b\sim\mathsf{Uni}(\{0,1\}^m)}[|h_0(b)| \le \alpha] \ge 1-\beta$ for α and β. Then, there exists a $(t;\epsilon,\delta)$-differentially private protocol for computing g with (α,β)-utility.

Proof: Define U as

$$U = \{([\![b_1]\!]_i, \ldots, [\![b_m]\!]_i)_{i\in[n]} : (b_1, \ldots, b_m) \in \{0,1\}^m\}$$

and $h : U \to \mathbb{Z}_q$ as $h([\![b_1]\!], \ldots, [\![b_m]\!]) = h_0(b_1, \ldots, b_m)$. Then, Π_0 t-securely computes $h(\boldsymbol{s})$ from $\boldsymbol{s} \in U$. Furthermore, the setup assumption of Π can be removed since generating correlated randomness $\boldsymbol{s} \sim \mathsf{Uni}(U)$ can be done by Σ_{Bit}. Due to the security of Σ_{Bit}, the conditional distribution of $h(\boldsymbol{s})$ given $\boldsymbol{s}_T$ is the same as $h_0(b)$ for $b \sim \mathsf{Uni}(\{0,1\}^m)$ as long as $|T| \le t$. Therefore, for any subset T of size t and any pair of T-neighboring vectors $\boldsymbol{x}, \boldsymbol{y} \in D^n$, two conditional distributions $(\boldsymbol{x}_T, \boldsymbol{s}_T, g(\boldsymbol{x})+h(\boldsymbol{s})), (\boldsymbol{y}_T, \boldsymbol{s}_T, g(\boldsymbol{y})+h(\boldsymbol{s}))$ given $\boldsymbol{s}_T$ are (ϵ,δ)-DP close for $\boldsymbol{s} \sim \mathsf{Uni}(U)$. The statements then follow from Proposition 1. □

The protocol provided in the above proof can also be viewed as a protocol securely realizing the randomized functionality which, on input $\boldsymbol{x} \in D^n$, outputs $g(\boldsymbol{x}) + h_0(b)$ for $b \sim \mathsf{Uni}(\{0,1\}^m)$. Thus, the same statement is obtained by applying the general result [8], [9] which states that protocols securely realizing differentially private mechanisms achieve the same level of differential privacy. Nevertheless, an important feature of Corollary 1 is that it reduces generation of noise for differential privacy to generation of *uniformly random* bits and secure computation of the *deterministic* function h_0.

4 Two Novel Noise Generation Protocols for the Discrete Laplace Distribution

To begin with, we present simple sufficient conditions for a probability distribution over a finite interval of $\mathbb{Z}$ to provide differential privacy.

Lemma 1. Let $N \in \mathbb{N}$ and $g : D^n \to \mathbb{Z}_q$ be a function with sensitivity Δ and range R. Assume that $q/2 > N + R$. Let X be a random variable on $\mathbb{Z}_q$ such that

1) its support is $\{z \in \mathbb{Z} : -N < z < N\}$;
2) $\Pr[X = z] = \Pr[X = -z]$ for $0 \le z < N$;
3) $\Pr[X = z] \le e^{\epsilon} \Pr[X = z']$ for $0 \le z \le z' \le z + \Delta$;
4) $\Pr[N - \Delta \le X < N] \le \delta$.

Then, $\mathcal{M}(\cdot) = g(\cdot) + X$ is (ϵ, δ)-differentially private.

Proof: Let $S \subseteq \mathbb{Z}_q$ and $\boldsymbol{x}, \boldsymbol{y} \in D^n$ be neighboring vectors. Assume that $g(\boldsymbol{y}) \le g(\boldsymbol{x})$. Define $S_1, S_2 \subseteq S$ as $S_1 = \{z \in S : g(\boldsymbol{x}) - N \le z \le g(\boldsymbol{y}) + N\}$ and $S_2 = S \setminus S_1$. Since $g(\boldsymbol{x}) \le g(\boldsymbol{y}) + \Delta$, we have

$$\begin{aligned}
&\Pr[\mathcal{M}(\boldsymbol{x}) \in S] \\
&= \Pr[\mathcal{M}(\boldsymbol{x}) \in S_1] + \Pr[\mathcal{M}(\boldsymbol{x}) \in S_2] \\
&\le \sum_{z \in S_1} \Pr[X = z - g(\boldsymbol{x})] + \Pr[N - \Delta \le X < N] \\
&\le \sum_{z \in S_1} e^{\epsilon} \Pr[X = z - g(\boldsymbol{y})] + \delta \\
&\le e^{\epsilon} \Pr[\mathcal{M}(\boldsymbol{y}) \in S] + \delta.
\end{aligned}$$

A similar argument works when $g(\boldsymbol{y}) \ge g(\boldsymbol{x})$. □

The output distributions of our protocols have some statistical difference from target distributions and our former protocol fails to generate shares with small but non-zero probability. We analyze the loss of differential privacy and utility in that regard.

Lemma 2. Let X, X' be random variables on $\mathbb{Z}_q$ such that $\mathsf{SD}(X, X') \le \delta_0$. Assume that $\mathcal{M}(\cdot) = g(\cdot) + X$ satisfies (ϵ, δ)-differential privacy and (α, β)-utility for g. Then, $\mathcal{M}'(\cdot) = g(\cdot) + X'$ satisfies (ϵ, δ')-differential privacy and $(\alpha, \beta + \delta_0)$-utility for g, where $\delta' = \delta_0(e^{\epsilon} + 1) + \delta$.

Proof: Let $S \subseteq \mathbb{Z}_q$ and $\boldsymbol{x}, \boldsymbol{y}$ be neighboring vectors. Then, it holds that

$$\begin{aligned}
&\Pr[\mathcal{M}'(\boldsymbol{x}) \in S] \\
&\le \Pr[\mathcal{M}(\boldsymbol{x}) \in S] + \delta_0 \\
&\le e^{\epsilon} \Pr[\mathcal{M}(\boldsymbol{y}) \in S] + \delta + \delta_0 \\
&\le e^{\epsilon} \Pr[\mathcal{M}'(\boldsymbol{y}) \in S] + \delta_0(e^{\epsilon} + 1) + \delta.
\end{aligned}$$

Furthermore, $(\alpha, \beta + \delta_0)$-utility follows from

$$\begin{aligned}
\Pr[|\mathcal{M}'(\boldsymbol{x}) - g(\boldsymbol{x})| \le \alpha] &= \Pr[|X'| \le \alpha] \\
&\ge \Pr[|X| \le \alpha] - \delta_0 \\
&\ge 1 - \beta - \delta_0.
\end{aligned}$$

□

Lemma 3. Let $\perp$ be a special symbol not in $\mathbb{Z}_q$. Let X be a random variable on $\mathbb{Z}_q \cup \{\perp\}$ and $\delta_0 = \Pr[X = \perp]$. Let X' be the random variable associated with the conditional distribution of X given $X \neq \perp$. Assume that $\mathcal{M}'(\cdot) = g(\cdot) + X'$ satisfies (ϵ, δ)-differential privacy and (α, β)-utility for g. Then, $\mathcal{M}(\cdot) = g(\cdot) + X$ satisfies (ϵ, δ)-differential privacy and $(\alpha, \beta + \delta_0)$-utility for g, where for any input $\boldsymbol{x}$, we define $\mathcal{M}(\boldsymbol{x}) = \perp$ if $X = \perp$.

Proof: Let $S \subseteq \mathbb{Z}_q \cup \{\perp\}$ and $\boldsymbol{x}, \boldsymbol{y}$ be neighboring vectors. Then, it holds that

$$\begin{aligned}
&\Pr[\mathcal{M}(\boldsymbol{x}) \in S] \\
&= \Pr[\mathcal{M}(\boldsymbol{x}) \neq \perp] \cdot \Pr[\mathcal{M}(\boldsymbol{x}) \in S \mid \mathcal{M}(\boldsymbol{x}) \neq \perp] \\
&\quad + \Pr[\mathcal{M}(\boldsymbol{x}) = \perp] \cdot \Pr[\mathcal{M}(\boldsymbol{x}) \in S \mid \mathcal{M}(\boldsymbol{x}) = \perp] \\
&\le (1 - \delta_0)(e^{\epsilon} \Pr[\mathcal{M}(\boldsymbol{y}) \in S \mid \mathcal{M}(\boldsymbol{y}) \neq \perp] + \delta) + \delta_0 \mathbb{1}_S(\perp) \\
&= e^{\epsilon}(\Pr[\mathcal{M}(\boldsymbol{y}) \in S] - \delta_0 \mathbb{1}_S(\perp)) + (1 - \delta_0)\delta + \delta_0 \mathbb{1}_S(\perp) \\
&= e^{\epsilon} \Pr[\mathcal{M}(\boldsymbol{y}) \in S] + (1 - \delta_0)\delta + (1 - e^{\epsilon})\delta_0 \mathbb{1}_S(\perp) \\
&\le e^{\epsilon} \Pr[\mathcal{M}(\boldsymbol{y}) \in S] + \delta,
\end{aligned}$$

where $\mathbb{1}_S(\perp)$ is 1 if $\perp \in S$ and 0 otherwise. Furthermore, $(\alpha, \beta + \delta_0)$-utility follows from

$$\begin{aligned}
\Pr[|\mathcal{M}(\boldsymbol{x}) - g(\boldsymbol{x})| \le \alpha] &= \Pr[|X| \le \alpha] \\
&= \Pr[X \neq \perp] \cdot \Pr[|X'| \le \alpha] \\
&\ge (1 - \delta_0)(1 - \beta) \\
&\ge 1 - \beta - \delta_0.
\end{aligned}$$

□

4.1 The First Protocol

As a building block, we present a constant-round protocol for generating noise according to the Bernoulli distribution from many uniformly random bits. Let $\mathsf{Ber}(\alpha)$ be the distribution over $\{0,1\}$ such that $\Pr[X = b] = \alpha^b(1-\alpha)^{1-b}$. For $\ell \in [d]$, let α_ℓ be the ℓ-th most significant bit in the binary expansion of α. Define $B_{\alpha,d} : \{0,1\}^d \to \mathbb{Z}_q$ as $B_{\alpha,d}(b_1, \ldots, b_d) = 1 - b_j$ for $j = \min\{\ell \in [d] : b_\ell \neq \alpha_\ell\}$ (we set $j = d + 1$ if there is no such index). Equivalently, it outputs 1 if $\sum_{\ell \in [d]} b_\ell 2^{-\ell} \le \alpha$ and otherwise, outputs 0. The statistical distance between $\mathsf{Ber}(\alpha)$ and $B_{\alpha,d}(b)$ for $b \sim \mathsf{Uni}(\{0,1\}^d)$ is at most 2^{-d}. The protocol $\textsc{Ber}_{\alpha,d}$ described in Fig. 2 t-securely computes shares of $B_{\alpha,d}(b)$ from $b \in \{0,1\}^d$. Note that at Step 1, $[\![b_\ell \oplus \alpha_\ell]\!]$ can be locally computed from $[\![b_\ell]\!]$ since α_ℓ is a public parameter.

Now, we construct a protocol for generating noise drawn from a finite-range variant of $\mathsf{DL}(p)$. Let $0 < p < 1$ and $N \in \mathbb{N}$. Let $G \sim \mathsf{Geo}(p, N)$ be the truncated geometric distribution defined by $\Pr[G = x] = C(1-p)p^x$ for $x \in [0..N)$, where $C = (\sum_{x \in [0..N)}(1-p)p^x)^{-1} = (1 - p^N)^{-1}$ is a normalizing constant. Define $\widetilde{\mathsf{FDL}}_1(p, N)$ as the distribution

Input. $[\![b_\ell]\!]$, $\ell \in [d]$, where $b_\ell \in \{0,1\}$.
Output. $[\![B_{\alpha,d}(b_1,\dots,b_d)]\!]$.
Protocol.

1) $[\![c_\ell]\!] = [\![b_\ell \oplus \alpha_\ell]\!]$ for $\ell \in [d]$.
2) $([\![e_\ell]\!])_{\ell\in[d]} = \mathrm{PRE}^d_\vee(([\![c_k]\!])_{k\in[d]})$.
3) $[\![f_\ell]\!] = [\![e_\ell]\!] - [\![e_{\ell-1}]\!]$ for $\ell \in [d]$, where $e_0 = 0$.
4) $[\![g]\!] = \mathrm{IP}^d(([\![f_\ell]\!])_{\ell\in[d]}, ([\![b_\ell]\!])_{\ell\in[d]})$.
5) Output $1 - [\![g]\!]$.

Fig. 2: The protocol $\mathrm{BER}_{\alpha,d}$ for computing $B_{\alpha,d}$

of $X_1 - X_2$ for independent $X_1, X_2 \sim \mathsf{Geo}(p,N)$. Explicitly, the distribution of $X' \sim \widetilde{\mathsf{FDL}}_1(p,N)$ is

$$\Pr[X'=x] = \frac{(1-p)p^{|x|}(1-p^{2(N-|x|)})}{(1+p)(1-p^N)^2}, \ -N<x<N.$$

The motivation behind this definition is a mathematical fact that the distribution of $g_1 - g_2$ is $\mathsf{DL}(p)$ for two independent samples g_1, g_2 drawn from the geometric distribution [40].

To make it applicable to a wide range of privacy budgets, for $M \in \mathbb{N}$ with $M < N$, we define $\mathsf{FDL}_1(p,N,M)$ as the probability distribution of $X' \sim \widetilde{\mathsf{FDL}}_1(p,N)$ conditioned on $|X'| \le M$. That is, $X \sim \mathsf{FDL}_1(p,N,M)$ has the probability distribution

$$\Pr[X=x] = \frac{\Pr[X'=x]}{\Pr[|X'|\le M]}, \ -M \le x \le M,$$

where $X' \sim \widetilde{\mathsf{FDL}}_1(p,N)$.

Proposition 2. Let $X \sim \mathsf{FDL}_1(p,N,M)$. The functionality $\mathcal{M}(\cdot) = g(\cdot) + X$ is (ϵ,δ)-differentially private for a function g with sensitivity Δ and range R if p, N, and M satisfies that $N + R < q/2$,

$$p^{-\Delta}\frac{1-p^{2(N-M)}}{1-p^{2(N-M-\Delta)}} \le e^\epsilon, \tag{2}$$

$$\text{and } \frac{p^{M-\Delta}}{(1-p)(1-p^{2N})} \le \delta. \tag{3}$$

Furthermore, $\mathcal{M}$ satisfies (α,β)-utility for g if $\alpha = (\Delta/\epsilon)\ln(2/(\beta(1-p)(1-p^{2N})))$.

Proof: The support of $\mathsf{FDL}_1(p,N,M)$ is $\{z \in \mathbb{Z} : |z| < M\}$. For any z and z' with $0 \le z \le z' \le z+\Delta$, we have

$$\begin{aligned}\frac{\Pr[X=z]}{\Pr[X=z']} &\le p^{-\Delta}\frac{1-p^{2(N-z)}}{1-p^{2(N-z-\Delta)}}\\ &\le p^{-\Delta}\frac{1-p^{2(N-M)}}{1-p^{2(N-M-\Delta)}}\end{aligned}$$

since $(1-p^{2(N-z)})/(1-p^{2(N-z-\Delta)})$ is monotonically increasing with respect to z.

To show $\Pr[M-\Delta \le X \le M] \le \delta$, we observe that for $X' \sim \widetilde{\mathsf{FDL}}_1(p,N)$ and any $0 \le m \le M$,

$$\begin{aligned}&\Pr[|X'|\le m]\\ &= \frac{1-p}{(1+p)(1-p^N)^2}\left(\sum_{|x|\le m} p^{|x|} - \sum_{|x|\le m} p^{2N-|x|}\right)\\ &= \frac{1+p+p^{2N}+p^{2N+1}-2p^{m+1}-2p^{2N-m}}{(1+p)(1-p^N)^2}.\end{aligned}$$

Thus, we have that

$$\begin{aligned}&\Pr[M-\Delta \le X \le M]\\ &= \frac{1}{2}\left(1 - \frac{\Pr[|X'|\le M-\Delta-1]}{\Pr[|X'|\le M]}\right)\\ &= \frac{p^{M-\Delta}(1-p^{\Delta+1})(1-p^{2N-2M+\Delta})}{1+p+p^{2N}+p^{2N+1}-2p^{M+1}-2p^{2N-M}}\\ &= \frac{p^{M-\Delta}(1-p^{\Delta+1})(1-p^{2N-2M+\Delta})}{(1+p)(1-p^N)^2\Pr[|X'|\le M]}\\ &\le \frac{p^{M-\Delta}}{(1+p)(1-p^N)^2\Pr[X'=0]}\\ &= \frac{p^{M-\Delta}}{(1-p)(1-p^{2N})}\\ &\le \delta.\end{aligned}$$

Differential privacy then follows from Lemma 1.

For $0 \le \alpha \le M$ and $k = \lfloor \alpha \rfloor$, we have

$$\begin{aligned}\Pr[|X|>\alpha] &= 1 - \frac{\Pr[|X'|\le k]}{\Pr[|X'|\le M]}\\ &= \frac{2p^{k+1}(1-p^{M-k})(1-p^{2N-M-k-1})}{(1+p)(1-p^N)^2\Pr[|X'|\le M]}\\ &< \frac{2p^{k+1}}{(1-p)(1-p^{2N})}.\end{aligned}$$

Therefore, $\mathcal{M}$ satisfies (α,β)-utility if $\beta = 2p^{\alpha+1}/(1-p)(1-p^{2N})$, i.e.,

$$\alpha = \frac{\Delta}{\epsilon}\ln\frac{2}{\beta(1-p)(1-p^{2N})}.$$

□

Remark. We give an explicit way to choose parameters p, N, M satisfying the conditions (2) and (3) given ϵ, δ, Δ. We first show a more simple sufficient condition for (2) that

$$K := 2(N-M-\Delta) \ge \max\left\{\frac{2}{\epsilon}, \frac{2(1-\zeta)}{\zeta^2}+1\right\} \tag{4}$$

$$\text{and } p = \exp\left(-\frac{\epsilon-\theta(K)}{\Delta}\right), \tag{5}$$

where $\zeta = \epsilon/2\Delta$ and $\theta(K) = \ln(1+K^{-1})$. Given ϵ,δ,Δ, choose K as any even number satisfying (4) and set p according to (5). Then, choose M as a sufficiently large number satisfying

$$\frac{p^{M-\Delta}}{(1-p)(1-p^{K+2M+2\Delta})} \le \delta$$

and finally set $N = K/2 + M + \Delta$.

To see (4) and (5) imply (2), let $\eta = \exp(\zeta)$. Since $\eta - 1 \ge \zeta$, we have that

$$\begin{aligned}\eta^K &= (1+(\eta-1))^K\\ &\ge 1 + K\zeta + \frac{K(K-1)}{2}\zeta^2\\ &\ge K+1.\end{aligned}$$

The last inequality follows from $K \ge 2(1-\zeta)/\zeta^2 + 1$. Since $K \ge 2/\epsilon$, we have $\epsilon - \theta(K) \ge \epsilon/2$ and

$$\exp\left(\frac{K(\epsilon-\theta(K))}{\Delta}\right) \ge \eta^K \ge K+1,$$

and hence

$$1-\exp\left(\frac{K(\epsilon-\theta(K))}{\Delta}\right)\geq\frac{K}{K+1}.$$

Since $\ln(K/(K+1)) = -\theta(K)$, we then have that

$$\begin{aligned}0&\leq\theta(K)+\ln\left(1-\exp\left(-\frac{K(\epsilon-\theta(K))}{\Delta}\right)\right)\\&=\theta(K)+\ln(1-p^K)\\&=\epsilon+\Delta\ln p+\ln(1-p^K).\end{aligned}$$

Hence, it holds that $\epsilon \geq -\Delta \ln p - \ln(1-p^K) + \ln(1-p^{2N})$ and $e^\epsilon \geq p^{-\Delta}(1-p^{2N})/(1-p^{2(N-\Delta)})$.

To obtain a noise generator function for $\mathsf{FDL}_1(p,N,M)$, we show the following lemmas. Lemma 4 readily follows from the definitions of $\widetilde{\mathsf{FDL}}_1(p,N)$ and $\mathsf{FDL}_1(p,N,M)$. Lemma 5 is implicitly shown in [5].

Lemma 4. Let $0<p<1$ and $N,M\in\mathbb{N}$ be such that $N>M$. Let X_1, X_2 be independent random variables following $\mathsf{Geo}(p,N)$. Define X over $\mathbb{Z}_q\cup\{\perp\}$ as

$$X=\begin{cases}X_1-X_2, & \text{if } |X_1-X_2|\leq M,\\ \perp, & \text{otherwise.}\end{cases}$$

Then, the distribution of X conditioned on $X\neq\perp$ is $\mathsf{FDL}_1(p,N,M)$. Furthermore, $\Pr[X=\perp]\leq 2p^{M+1}/((1+p)(1-p^N)^2)$.

Lemma 5 ([5]). Let $N=2^c$ for $c\in\mathbb{N}$, $0<p<1$, and $\beta_i=(1+p^{2^{-i}})^{-1}$ for $i\in[0..c)$. Define $G_{p,N}:\{0,1\}^c\to\mathbb{Z}_q$ as $G_{p,N}(b_0,\dots,b_{c-1})=\sum_{i\in[0..c)}b_i 2^i$. If $X_i\sim\mathsf{Ber}(\beta_i)$ for each $i\in[0..c)$, then $G_{p,N}(X_0,\dots,X_{c-1})$ follows $\mathsf{Geo}(p,N)$.

Proof: Define $\mathbb{Z}_{\geq c}=\{z\in\mathbb{Z}: z\geq c\}$ for $c\in\mathbb{Z}$. Let $\beta_i=(1+p^{-2^i})^{-1}$ also for $i\in\mathbb{Z}_{\geq c}$. It is shown in [5] that $G:=\sum_{i\in\mathbb{Z}_{\geq 0}}X_i 2^i$ has the geometric distribution $\mathsf{Geo}(p)$, i.e., $\Pr[G=g]=(1-p)p^g$ for $g\geq 0$. Therefore, for every finite set $I\subseteq\mathbb{Z}_{\geq 0}$,

$$\Pr[X_i=1\ (\forall i\in I)\wedge X_i=0\ (\forall i\in(\mathbb{Z}_{\geq 0}\setminus I))]=(1-p)p^{y(I)},$$

where we define $y(I)=\sum_{i\in I}2^i$. Let $g\in[0..N)$ and $I\subseteq[0..c)$ be such that $g=y(I)$.

$$\begin{aligned}\Pr\left[\sum_{i\in[0..c)}X_i 2^i=g\right]&=\Pr\left[\begin{array}{l}X_i=1\ (\forall i\in I)\wedge\\ X_i=0\ (\forall i\in([0..c)\setminus I))\end{array}\right]\\&=\sum_{J\subseteq\mathbb{Z}_{\geq c}}(1-p)p^{y(I\cup J)}\\&=(1-p)p^g\sum_{J\subseteq\mathbb{Z}_{\geq c}}p^{y(J)}.\end{aligned}$$

Since $\sum_{g\in[0..N)}\Pr\left[\sum_{i\in[0..c)}X_i 2^i=g\right]=1$, we have $\sum_{J\subseteq\mathbb{Z}_{\geq c}}p^{y(J)}=(\sum_{g\in[0..N)}(1-p)p^g)^{-1}=C$. □

Now, let $c,d\in\mathbb{N}$, $N=2^c$, $M<N$, $m=2cd$, and $0<p<1$. Define $\beta_i=(1+p^{2^{-i}})^{-1}$ for $i\in[0..c)$. Define $F_1:\{0,1\}^m\to\mathbb{Z}_q\cup\{\perp\}$ as follows: For $\boldsymbol{u}=((u_{i1},\dots,u_{id})_{i\in[0..c)},(v_{i1},\dots,v_{id})_{i\in[0..c)})\in\{0,1\}^m$,

1) for $i\in[0..c)$ and $j\in[d]$, let $a_i=B_{\beta_i,d}(u_{i1},\dots,u_{id})$ and $b_i=B_{\beta_i,d}(v_{i1},\dots,v_{id})$;
2) let $x_1=G_{p,N}(a_0,\dots,a_{c-1})$ and $x_2=G_{p,N}(b_0,\dots,b_{c-1})$;
3) let

$$F_1(\boldsymbol{u})=\begin{cases}x_1-x_2, & \text{if } |x_1-x_2|\leq M,\\ \perp, & \text{otherwise.}\end{cases}$$

Input. $([\![a_{i1}]\!],\dots,[\![a_{id}]\!])_{i\in[0..c)},([\![b_{i1}]\!],\dots,[\![b_{id}]\!])_{i\in[0..c)}$, where $a_{ij},b_{ij}\in\{0,1\}$.
Output. $[\![F_1((a_{i1},\dots,a_{id})_{i\in[0..c)},(b_{i1},\dots,b_{id})_{i\in[0..c)})]\!]$.
Protocol.

1) $[\![a_i]\!]=\textsc{Ber}_{\beta_i,d}(([\![a_{ij}]\!])_{j\in[d]})$ for $i\in[0..c)$.
2) $[\![b_i]\!]=\textsc{Ber}_{\beta_i,d}(([\![b_{ij}]\!])_{j\in[d]})$ for $i\in[0..c)$.
3) $[\![x_1]\!]=\textsc{Geo}_{p,N}(([\![a_i]\!])_{i\in[0..c)})$.
4) $[\![x_2]\!]=\textsc{Geo}_{p,N}(([\![b_i]\!])_{i\in[0..c)})$.
5) $[\![y]\!]=[\![x_1]\!]-[\![x_2]\!]$.
6) $[\![e]\!]=[\![|y|\overset{?}{\leq}M]\!]$ and open e.
7) Output $[\![y]\!]$ if $e=1$ and otherwise, output $\perp$.

Fig. 3: The protocol Π_{FDL_1} for computing F_1

Since the statistical distance between a_i (or b_i) and $\mathsf{Ber}(\beta_i)$ is at most 2^{-d}, we have that $\mathsf{SD}(F_1(\boldsymbol{u}),X)\leq c2^{-d+1}$ for $\boldsymbol{u}\sim\mathsf{Uni}(\{0,1\}^m)$ and X defined in Lemma 4.

In view of Lemmas 2, 3, and Proposition 2, we can estimate differential privacy and utility provided by $F_1(\boldsymbol{u})$ for $\boldsymbol{u}\sim\mathsf{Uni}(\{0,1\}^m)$. We show an MPC protocol for F_1 in Fig. 3. Combining it with Corollary 1, we have the following theorem. Note that since $G_{p,N}$ is a linear function of inputs, it is straightforward to construct a non-interactive protocol $\textsc{Geo}_{p,N}$ securely computing $G_{p,N}$.

Theorem 1. Let $g:D^n\to\mathbb{Z}_q$ be a function with sensitivity Δ and range R. For $\epsilon>0$ and $\delta>0$, let $0<p<1$, $N=2^c$ for $c\in\mathbb{N}$, and $M\in\mathbb{N}$ be the ones satisfying the conditions (2), (3), and that $q/2>N+R$. Let $\alpha,\beta\in\mathbb{R}$ be such that $\alpha=(\Delta/\epsilon)\ln(2/(\beta(1-p)(1-p^{2N})))$. For any $t<n/2$ and any $d\in\mathbb{N}$, there is a $(t;\epsilon,\delta')$-differentially private protocol computing g with $(\alpha,\beta+\delta_0+\delta_1)$-utility, where $\delta_0=c2^{-d+1}$, $\delta_1=2p^{M+1}/((1+p)(1-p^N)^2)$, and $\delta'=\delta+\delta_0(e^\epsilon+1)$.

4.2 The Second Protocol

Next, we construct a protocol for generating noise drawn from another finite-range variant of $\mathsf{DL}(p)$. Let $0<p<1$ and $N\in\mathbb{N}$. Let $\mathsf{FDL}_2(p,N)$ denote the probability distribution over $\{z\in\mathbb{Z}:-N\leq z\leq N\}$ defined as

$$\Pr[X=x]=\begin{cases}p^{|x|}(1-p)/(1+p), & \text{if } |x|<N,\\ p^N/(1+p), & \text{if } |x|=N,\\ 0, & \text{otherwise.}\end{cases}$$

Proposition 3. Let $X\sim\mathsf{FDL}_2(p,N)$. The functionality $\mathcal{M}(\cdot)=g(\cdot)+X$ is (ϵ,δ)-differentially private for a function $g:D^n\to\mathbb{Z}_q$ with sensitivity Δ and range R if

$$p=\exp\left(-\frac{\epsilon}{\Delta}\right),\ p^N\frac{1+p^{-\Delta}}{1+p}\leq\delta,\ \text{and}\ \frac{q}{2}>N+R. \tag{6}$$

Furthermore, $\mathcal{M}$ satisfies (α, β)-utility for g if $\alpha = (\Delta/\epsilon)\ln(2/\beta)$.

Proof: Differential privacy roughly follows by applying Lemma 1 to $\mathsf{FDL}_2(p, N)$ although we have to slightly modify Lemma 1 so that $\Pr[N - \Delta \le X \le N]$ (instead of $\Pr[N - \Delta \le X < N]$) is upper bounded by δ. For $0 \le \alpha < N$ and $k = \lfloor \alpha \rfloor$, we have

$$\Pr[|X| \le \alpha] = \sum_{i=-k}^{k} \frac{1-p}{1+p} p^i = 1 - \frac{2p^{k+1}}{1+p} > 1 - \frac{2p^\alpha}{1+p}.$$

Therefore, $\mathcal{M}$ satisfies (α, β)-utility if $\beta = 2p^\alpha/(1+p)$, i.e.,

$$\alpha = \frac{\Delta}{\epsilon} \ln \frac{2}{\beta(1+p)} \le \frac{\Delta}{\epsilon} \ln \frac{2}{\beta}.$$

□

We show that sampling a value from $\mathsf{FDL}_2(p, N)$ can be reduced to sampling sufficiently many bits from $\mathsf{Ber}(\alpha)$.

Proposition 4. Assume that $q/2 > N$. Let B_0 be a random variable with $\mathsf{Ber}((1-p)/(1+p))$ and $B_1, \ldots, B_{N-1}$ be independent random variables with $\mathsf{Ber}(1-p)$. Define a random variable Y on $\mathbb{Z}_q$ as

$$Y = \begin{cases} \min\{i \in [0..N) : B_i = 1\}, & \text{if } B_i = 1 \text{ for some } i, \\ N, & \text{otherwise.} \end{cases}$$

Then, the distribution of $X = \sigma Y$ for $\sigma \sim \mathsf{Uni}(\{-1, +1\})$ is $\mathsf{FDL}_2(p, N)$.

Proof: Let $p_0 = (1-p)/(1+p)$ and $p_1 = 1-p$. Observe that $\Pr[\sigma Y = 0] = p_0$, $\Pr[\sigma Y = k] = (1/2)(1 - p_0)(1-p_1)^{|k|-1} p_1 = p^{|k|}(1-p)/(1+p)$ if $0 < |k| < N$, and $\Pr[\sigma Y = k] = (1/2)(1-p_0)(1-p_1)^N = p^N/(1+p)$ if $|k| = N$. □

We then obtain a noise generator function for $\mathsf{FDL}_2(p, N)$. Let $N, d \in \mathbb{N}$, $m = Nd + 1$, and $0 < p < 1$. Define γ_i for $i \in [0..N)$ as $\gamma_0 = (1-p)/(1+p)$ and $\gamma_i = 1 - p$ for $i \neq 0$. Define $F_2 : \{0,1\}^m \to \mathbb{Z}_q$ as follows: For $\boldsymbol{b} = ((b_{i1}, \ldots, b_{id})_{i \in [0..N)}, b) \in \{0,1\}^m$,

1) let $b_i = B_{\gamma_i, d}(b_{i1}, \ldots, b_{id})$ for $i \in [0..N)$ and $\sigma = 1 - 2b$;
2) let $y = \min\{i \in [0..N) : b_i = 1\}$ if $b_i = 1$ for some i and otherwise $y = N$;
3) let $F_2(\boldsymbol{b}) = \sigma y$.

Since the statistical distance between b_i and $\mathsf{Ber}(\gamma_i)$ is at most 2^{-d}, we have that $\mathsf{SD}(F_2(\boldsymbol{b}), \mathsf{FDL}_2(p, N)) \le N2^{-d}$ for $\boldsymbol{b} \sim \mathsf{Uni}(\{0,1\}^m)$. We can also directly obtain an MPC protocol Π_{FDL_2} for F_2 (Fig. 4). Combining it with Corollary 1, we have the following theorem.

Theorem 2. Let $g : D^n \to \mathbb{Z}_q$ be a function with sensitivity Δ and range R. For $\epsilon > 0$ and $\delta > 0$, let $0 < p < 1$ and $N \in \mathbb{N}$ be the ones satisfying the condition (6). Let $\alpha, \beta \in \mathbb{R}$ be such that $\alpha = (\Delta/\epsilon)\ln(2/\beta)$. For any $t < n/2$ and any $d \in \mathbb{N}$, there is a $(t; \epsilon, \delta')$-differentially private protocol computing g with $(\alpha, \beta + \delta_0)$-utility, where $\delta_0 = N2^{-d}$ and $\delta' = \delta + \delta_0(e^\epsilon + 1)$.

Input. $([\![b_{i1}]\!], \ldots, [\![b_{id}]\!])_{i \in [0..N)}, [\![b]\!]$, where $b_{ij}, b \in \{0,1\}$.
Output. $[\![F_2((b_{i1}, \ldots, b_{id})_{i \in [0..N)}, b)]\!]$.
Protocol.

1) $[\![b_i]\!] = \textsc{Ber}_{\gamma_i, d}(([\![b_{ij}]\!])_{j \in [d]})$ for $i \in [0..N)$.
2) $([\![c_i]\!])_{i \in [0..N)} = \textsc{Pre}^N_\vee(([\![b_j]\!])_{j \in [0..N)})$.
3) $[\![y]\!] = N - \sum_{i \in [0..N)} [\![c_i]\!]$.
4) $[\![\sigma]\!] = 1 - 2[\![b]\!]$.
5) Output $[\![\sigma y]\!] = \textsc{Mult}([\![y]\!], [\![\sigma]\!])$.

Fig. 4: The protocol Π_{FDL_2} for computing F_2

Public parameter. $a \in \{0,1\}^s$.
Input. $\boldsymbol{s} = ((r_A)_{A \in \mathcal{A}_i})_{i \in [n]} \in U$.
Output. $([\![h(\boldsymbol{s})]\!]_i)_{i \in [n]}$.
Protocol. Output $\ell_i = \sum_{A \in \mathcal{A}_i} \ell_{r_A}(a) f_A(i)$ for $i \in [n]$.

Fig. 5: The protocol Π_{Bin} for computing h

5 A Novel Noise Generation Protocol for the Binomial Distribution

We provide a protocol which allows parties to non-interactively obtain a share of noise drawn from $\mathsf{NBin}(\ell, M)$ by using predistributed keys for a pseudorandom function.

We define some notations. Let $\lambda \in \mathbb{N}$ be a security parameter. Let U be the set of all possible tuples of keys for pseudorandom secret sharing. Formally, recall that we have defined $\mathcal{A} = \{A \subseteq [n] : |A| = n - t\}$ and $\mathcal{A}_i = \{A \in \mathcal{A} : i \in A\}$ for $i \in [n]$. We define $S = \{0,1\}^{\lambda \cdot \binom{n-1}{t}}$ and

$$U = \left\{ ((r_{A,i})_{A \in \mathcal{A}_i})_{i \in [n]} \in S^n : \begin{array}{l} r_{A,i} = r_{A,j} \text{ for all } i, j, A \\ \text{with } A \in \mathcal{A}_i \cap \mathcal{A}_j \end{array} \right\}.$$

We restrict ourselves to the family $\{\psi_r : \{0,1\}^{s(\lambda)} \to \{0,1\}^{\ell(\lambda)}\}_{r \in \{0,1\}^\lambda}$ with key length of λ and simply write $s = s(\lambda)$ and $\ell = \ell(\lambda)$. Let $\ell_r(a)$ be the number of 1's in $\psi_r(a) \in \{0,1\}^\ell$ for $r \in \{0,1\}^\lambda$ and $a \in \{0,1\}^s$.

Assume that $q > \ell|\mathcal{A}|$. Since $\boldsymbol{s} \in U$ can be written as $\boldsymbol{s} = ((r_A)_{A \in \mathcal{A}_i})_{i \in [n]}$ for some $r_A \in \{0,1\}^\lambda$, it is possible to define a function $h : U \to \mathbb{Z}_q$ (depending on a public input $a \in \{0,1\}^s$) as $h(\boldsymbol{s}) = \sum_{A \in \mathcal{A}} \ell_{r_A}(a)$. Then, the pseudorandomness of ψ_r implies that $(1/M)(h(\boldsymbol{s}) - \ell|\mathcal{A}|/2)$ works as a noise generator function for $\mathsf{NBin}(\ell|\mathcal{A}|, M)$ in the computational setting. The protocol Π_{Bin} described in Fig. 5 non-interactively (and hence t-securely) computes shares of $h(\boldsymbol{s})$ from $\boldsymbol{s} \in U$. Note that generating a uniformly random element in U is equivalent to generating random shares of the replicated secret sharing scheme, which in turn can be securely done by a known protocol Σ_{Rep} [36].

The following theorem (Theorem 3) mostly follows from the above observation and Proposition 1. One exception is that parties run Π_{Mg} rather than Π_g, where Mg is defined as $(Mg)(\boldsymbol{x}) = M \cdot g(\boldsymbol{x})$, and then compute $(1/M)(y - \ell|\mathcal{A}|/2)$ from the recovered secret y as plaintexts. We do that procedure to avoid an arithmetic operation over $\mathbb{R}$ in secret-shared form. For completeness, we formally describe the

Public parameter. $a \in \{0,1\}^s$.
Input. $\boldsymbol{x} \in D^n$.
Output. $z = g(\boldsymbol{x}) + (1/M)(h(\boldsymbol{s}) - \ell|\mathcal{A}|/2) \in \mathbb{R}$.
Protocol.
1) $[\![Mg(\boldsymbol{x})]\!] = \Pi_{Mg}(\boldsymbol{x})$.
2) $\boldsymbol{s} = \Sigma_{\text{Rep}}()$.
3) $[\![h(\boldsymbol{s})]\!] = \Pi_{\text{Bin}}(\boldsymbol{s})$.
4) $[\![y]\!] = [\![Mg(\boldsymbol{x})]\!] + [\![h(\boldsymbol{s})]\!]$.
5) Open y and output $z = (1/M)(y - \ell|\mathcal{A}|/2)$.

Fig. 6: The differentially private protocol Π based on the binomial distribution

protocol Π in Fig. 6. Its utility is discussed after the proof of Theorem 3.

Theorem 3. Let $g : D^n \to \mathbb{Z}_q$ be a function with sensitivity Δ and range R. Let λ be a security parameter and assume a pseudorandom function $\{\psi_r : \{0,1\}^s \to \{0,1\}^\ell\}_{r\in\{0,1\}^\lambda}$. For $\epsilon \in O(\log\lambda)$ and $\delta > 0$, assume that there exists $M \in \mathbb{N}$ such that $\epsilon \geq \epsilon(\delta, \ell, M, \Delta)$ and $q > MR + \ell|\mathcal{A}|$. For any $t < n/2$, the protocol Π described in Fig. 6 is a $(t;\epsilon,\delta)$-computationally differentially private protocol for computing g.

Proof: Fix a set $T \subseteq [n]$ of t corrupted parties and let $J = [n] \setminus T \in \mathcal{A}$. The output of Π has the form of $\mathcal{M}_g^{\ell|\mathcal{A}|,M}(\boldsymbol{x})$. However, the adversary knows all but one keys r_A, $A \neq J$ and the only noise unknown to him is $\ell_{r_J}(a)$. Therefore, an achievable level of differential privacy against the adversary's view deteriorates to that of $\mathcal{M}_g^{\ell,M}$.

More formally, let $f(\boldsymbol{x};\boldsymbol{s}) = g(\boldsymbol{x}) + (1/M)(h(\boldsymbol{s}) - \ell|\mathcal{A}|/2)$ for $\boldsymbol{x} \in D^n$ and $\boldsymbol{s} \in U$. In view of Proposition 1, it is sufficient to show that for all T-neighboring vectors $\boldsymbol{x}, \boldsymbol{y}$, two distributions $(\boldsymbol{x}_T, \boldsymbol{s}_T, f(\boldsymbol{x};\boldsymbol{s})), (\boldsymbol{y}_T, \boldsymbol{s}_T, f(\boldsymbol{y};\boldsymbol{s}))$ induced by $\boldsymbol{s} \sim \mathsf{Uni}(U)$ are computationally (ϵ,δ)-DP close even when the randomness of $\boldsymbol{s}_T$ is fixed. Recall that we have defined $U_T = \{\boldsymbol{u} \in S^{|T|} : \boldsymbol{u} = \boldsymbol{s}_T \text{ for some } \boldsymbol{s} \in U\}$.

First, define a randomized function $\widetilde{f}_J$ as $\widetilde{f}_J(\boldsymbol{x};\boldsymbol{u}) = f_J(\boldsymbol{x};\boldsymbol{u}) + (1/M)(\widetilde{\ell}_J - \ell/2)$ for $\boldsymbol{x} \in D^n$ and $\boldsymbol{u} = ((r_A)_{A\in\mathcal{A}_i})_{i\in T} \in U_T$, where $f_J(\boldsymbol{x};\boldsymbol{u}) = g(\boldsymbol{x}) + (1/M)\sum_{A\in\mathcal{A}\setminus\{J\}}(\ell_{r_A}(a) - \ell/2)$ and $\widetilde{\ell}_J \sim \mathsf{Bin}(\ell, 1/2)$. In other words, if $\boldsymbol{u} = \boldsymbol{s}_T$, $\widetilde{f}_J(\boldsymbol{x};\boldsymbol{u})$ is defined by replacing $\ell_{r_J}(a)$ in $f(\boldsymbol{x};\boldsymbol{s})$ with $\widetilde{\ell}_J$ properly sampled from $\mathsf{Bin}(\ell,1/2)$. For fixed $\boldsymbol{u} \in U_T$, $\widetilde{f}_J(\cdot;\boldsymbol{u})$ is equivalent to the (ϵ,δ)-differentially private mechanism $\mathcal{M}_{f_J(\cdot;\boldsymbol{u})}^{\ell,M}$ and hence for all T-neighboring vectors $\boldsymbol{x},\boldsymbol{y}$ and for any $\boldsymbol{u} \in U_T$, two distributions $(\boldsymbol{x}_T, \boldsymbol{u}, \widetilde{f}_J(\boldsymbol{x};\boldsymbol{u})), (\boldsymbol{y}_T, \boldsymbol{u}, \widetilde{f}_J(\boldsymbol{y};\boldsymbol{u}))$ are (ϵ,δ)-DP close, where the randomness is $\ell_j \sim \mathsf{Bin}(\ell,1/2)$.

Next, we show that for any but fixed $\boldsymbol{x} \in D^n$ and $\boldsymbol{u} = ((r_A)_{A\in\mathcal{A}_i})_{i\in T} \in U_T$, the distribution of $\widetilde{f}_J(\boldsymbol{x};\boldsymbol{u})$ induced by $\widetilde{\ell}_J \sim \mathsf{Bin}(\ell,1/2)$ and that of $f(\boldsymbol{x};((r_A)_{A\in\mathcal{A}_i})_{i\in[n]}) = f_J(\boldsymbol{x};\boldsymbol{u}) + (1/M)(\ell_{r_J}(a) - \ell/2)$ induced by $r_J \sim \mathsf{Uni}(\{0,1\}^\lambda)$ are computationally indistinguishable. Assume otherwise that there are a PPT distinguisher D which can distinguish $(\boldsymbol{x}_T, \boldsymbol{u}, f(\boldsymbol{x};((r_A)_{A\in\mathcal{A}_i})_{i\in[n]}))$ from $(\boldsymbol{x}_T, \boldsymbol{u}, \widetilde{f}_J(\boldsymbol{x};\boldsymbol{u}))$ with non-negligible advantage for some $\boldsymbol{x} \in D^n$ and $\boldsymbol{u} = ((r_A)_{A\in\mathcal{A}_i})_{i\in T} \in U_T$. We construct a PPT oracle machine Alg for $\{\psi_r : \{0,1\}^s \to \{0,1\}^\ell\}_{r\in\{0,1\}^\lambda}$ as follows. First, Alg invokes the oracle to receive $\xi^b \in \{0,1\}^\ell$, and sets ℓ_J^b as the number of 1's in ξ^b. Here, the oracle flips a bit $b \sim \mathsf{Uni}(\{0,1\})$ and sets $\xi^b = \psi_{r_J}(a)$ for $r_J \sim \mathsf{Uni}(\{0,1\}^\lambda)$ if $b = 1$ or else $\xi^b = F(a)$ for a uniformly selected map $F : \{0,1\}^s \to \{0,1\}^\ell$. Next, Alg computes $z^b = f_J(\boldsymbol{x};\boldsymbol{u}) + (1/M)\left(\ell_J^b - \ell/2\right)$. Note that z^0 (resp. z^1) has the same distribution as $\widetilde{f}_J(\boldsymbol{x};\boldsymbol{u})$ (resp. $f(\boldsymbol{x};((r_A)_{A\in\mathcal{A}_i})_{i\in[n]})$ with $r_J \sim \mathsf{Uni}(\{0,1\}^\lambda)$). Alg then gives $(\boldsymbol{x}_T, \boldsymbol{u}, z^b)$ to D and receives a guess $b' \in \{0,1\}$ from D. Finally, it outputs b'. We would have that $\Pr[b' = b] - 1/2$ is non-negligible, which contradicts the indistinguishability of ψ.

For any PPT distinguisher D, for all T-neighboring vectors $\boldsymbol{x},\boldsymbol{y}$, and for any $\boldsymbol{u} \in U_T$, we have that

$$
\begin{aligned}
&\Pr_{\boldsymbol{s}\sim\mathsf{Uni}(U)}[\mathsf{D}(\boldsymbol{x}_T, \boldsymbol{s}_T, f(\boldsymbol{x};\boldsymbol{s})) = 1 \mid \boldsymbol{s}_T = \boldsymbol{u}] \\
&= \Pr_{\ell_J\sim\mathsf{Bin}(\ell,1/2)}\left[\mathsf{D}(\boldsymbol{x}_T, \boldsymbol{u}, \widetilde{f}_J(\boldsymbol{x};\boldsymbol{u})) = 1\right] + \mathsf{negl}(\lambda) \\
&\leq e^\epsilon \Pr_{\ell_J\sim\mathsf{Bin}(\ell,1/2)}\left[\mathsf{D}(\boldsymbol{y}_T, \boldsymbol{u}, \widetilde{f}_J(\boldsymbol{y};\boldsymbol{u})) = 1\right] + \delta + \mathsf{negl}(\lambda) \\
&= e^\epsilon \Pr[\mathsf{D}(\boldsymbol{y}_T, \boldsymbol{s}_T, f(\boldsymbol{y};\boldsymbol{s})) = 1 \mid \boldsymbol{s}_T = \boldsymbol{u}] + \delta + \mathsf{negl}(\lambda)\,.
\end{aligned}
$$

Here, we use the assumption that $\epsilon \in O(\log\lambda)$. □

As for the utility, we assume that the statistical distance between the uniform distribution over $\{0,1\}^{\ell|\mathcal{A}|}$ and $(\psi_{r_A}(a))_{A\in\mathcal{A}}$ for $r_A \sim \mathsf{Uni}(\{0,1\}^\lambda)$ is at most δ_ψ. Then, the statistical distance between $\mathsf{Bin}(\ell|\mathcal{A}|, 1/2)$ and $\sum_{A\in\mathcal{A}}\ell_{r_A}(a)$ for $r_A \sim \mathsf{Uni}(\{0,1\}^\lambda)$ is also upper bounded by δ_ψ. Based on the utility of $\mathcal{M}_g^{\ell|\mathcal{A}|,M}$, we can estimate that the functionality $\mathcal{M}(\cdot) = g(\cdot) + (1/M)(\sum_{A\in\mathcal{A}}\ell_{r_A}(a) - \ell|\mathcal{A}|/2)$ and hence the protocol Π satisfy $(\alpha, \beta + \delta_\psi)$-utility for g if $\alpha = \sqrt{(\ell|\mathcal{A}|/2M^2)\ln(2/\beta)}$.

6 Comparison

6.1 The Discrete Laplace Distribution

We compare our protocols in Section 4 with the one in [5]. The following comparison is based on the MPC primitives in Section 2.2.3. To describe their protocol [5], let N be a power of 2 and $L \sim \mathsf{TDL}(p,N)$ be the truncated discrete Laplace distribution, i.e., $\Pr[L = k] = Cp^{|k|}(1-p)/(1+p)$, $-N < k < N$, where $C = (1+p)/(1+p-2p^N)$ is a normalizing constant. Technically, their protocol generates shares of g drawn from $\mathsf{Geo}(p,N)$ using $\log N$ independent biased bits. Then, according to [29], σg conditioned on $(g,\sigma) \neq (0,-1)$ follows $\mathsf{TDL}(p,N)$ if $\sigma \sim \mathsf{Uni}(\{-1,+1\})$. Since the statistical distance between the output distribution and $\mathsf{TDL}(p,N)$ is $2^{-d}\log N$ for a parameter d, it follows from Lemma 1 that to achieve (ϵ,δ)-differential privacy for a function with sensitivity Δ, it is necessary to choose p, N, d such that $p = \exp(-\epsilon/\Delta)$ and

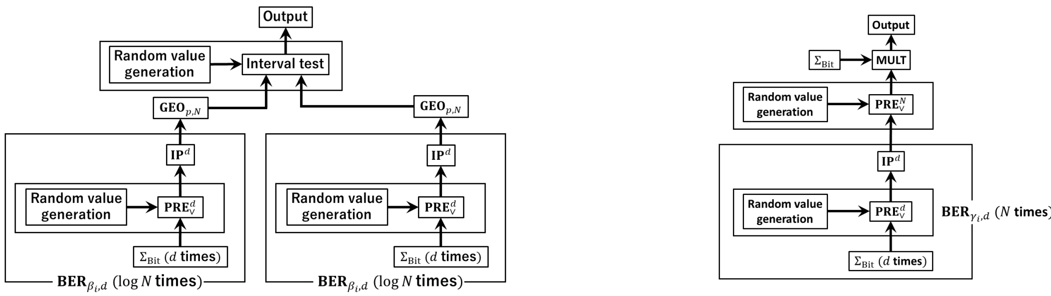
$$
(e^\epsilon + 1)2^{-d}\log N + \frac{p^N(p^{-\Delta} - 1)}{1 + p - 2p^N} \leq \delta. \tag{7}
$$

6.1.1 Round Complexity

By performing the sub-protocols in parallel as much as possible (Fig. 7), our protocols produce samples from $\mathsf{FDL}_1(p,N,M)$ and $\mathsf{FDL}_2(p,N)$ in 19 rounds and 14 rounds, respectively, which are independent of parameters p, N, or M. However, the protocol [5] needs to evaluate a certain

TABLE 1: Comparison of protocols sampling noise from finite-range discrete Laplace distributions

Reference	Distribution	Round	Communication	Probability of failure	Statistical distance
[5]	$\mathsf{TDL}(p, N)$	$2\lceil\log d\rceil + 11$	$O(d\log N + \log q)$	$(1-p)/2$	$2^{-d}\log N$
Theorem 1	$\mathsf{FDL}_1(p, N, M)$	19	$O(d\log N + \log q)$	$2p^{M+1}/((1+p)(1-p^N)^2)$	$2^{-d}\log N$
Theorem 2	$\mathsf{FDL}_2(p, N)$	14	$O(dN)$	0	$2^{-d}N$



(a) $\mathsf{FDL}_1(p, N, M)$ **(b)** $\mathsf{FDL}_2(p, N)$

Fig. 7: Our protocols for the discrete Laplace distributions performing the sub-protocols in parallel as much as possible

Boolean circuit for generating biased bits. The authors of [42] propose a circuit of size $7d - 3$ and depth $2\lceil\log d\rceil + 2$ to sample biased bits with statistical difference at most 2^{-d}. The total round complexity of [5] is therefore given as $2\lceil\log d\rceil + 11$, which depends on δ due to the condition (7).

We note that it is possible to further reduce the above round complexity by using pseudorandom secret sharing, which comes at the cost of satisfying only computational differential privacy. Specifically, our protocols can be done in 8 rounds for $\mathsf{FDL}_1(p, N, M)$ and 7 rounds for $\mathsf{FDL}_2(p, N)$ while [5] needs $2\lceil\log d\rceil + 7$ rounds by replacing the primitives in Section 2.2.3 with the ones based on pseudorandom secret sharing [39].

6.1.2 Utility

In view of Lemmas 2 and 3, a protocol achieves $(\alpha, \beta + \delta_0 + \delta_1)$-utility if its target noise distribution provides (α, β)-utility, the statistical distance from its output distribution is δ_0, and the probability of failure is δ_1. Since the probability of failure of [5] is $\delta_1 = (1-p)/2$ depending only on ϵ and Δ, it is impossible to make it negligible no matter how we choose other parameters d, N, and q. On the other hand, our protocols always generate an appropriate value or fail only with negligible probability. Hence, it is possible to make the utility of our protocols arbitrarily close to that of the target noise distribution.

6.1.3 Communication Complexity

It follows from the size of the Boolean circuit [42] that the protocol [5] requires $9d\log N - 3\log N + 162\log q + 4 = O(d\log N + \log q)$ invocations. Our protocol for $\mathsf{FDL}_1(p, N, M)$ needs $38d\log N + 2\log N + 110\log q + 1 = O(d\log N + \log q)$ invocations and the one for $\mathsf{FDL}_2(p, N)$ needs $19dN + 18N + 3 = O(dN)$ invocations. Although our protocols require asymptotically higher communication complexity than [5], the difference is not significant in practical parameter settings as described below.

6.1.4 Concrete Comparison

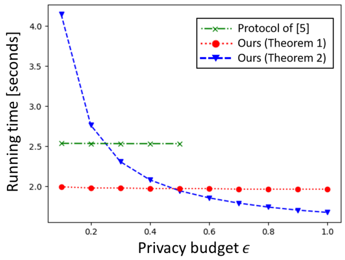
We estimate the running time required by the protocol of [5] and ours (Theorems 1 and 2). Consider the client-server model and set $n = 3$ and $t = 1$. Let g be any function with sensitivity $\Delta = 1$ and $q = 2^{61} - 1$. We assume the WAN setting, and set the network speed to $B = 100\,\mathrm{Mbits/sec}$ and the latency to $0.1\,\mathrm{sec}$ [43]. We calculate the running time as $(\#\,\text{comm. bits})/B + (\#\,\text{rounds}) \times (\text{latency})$, assuming that the local computation time is negligible compared with the communication time. Times to generate one Laplace noise are shown in Fig. 8, where the privacy budget ϵ ranges from 0.1 to 1, for

- $\delta = 2^{-100}$, 2^{-80} and 2^{-60}, with $\beta = 0.2$ and $\alpha = 50$;
- $\alpha = 10$, 20 and 30, with $\beta = 0.2$ and $\delta = 2^{-60}$.

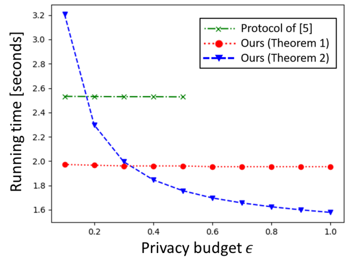
We do not plot the cases where protocols cannot achieve that privacy budget ϵ under the constraint of the (α, β)-utility.

We figure out that our protocols are up to around 1.3 times faster than the protocol [5]; e.g., our first protocol is at least 1.27 times faster in Fig. 8 (a). This is because the dominant cost in the running time comes from the round complexity in the WAN setting. Since the protocol [5] has non-negligible probability of failure $(1-p)/2 = (1 - \exp(-\epsilon/\Delta))/2$, it cannot be applied to some ranges of ϵ under the constraint of utility. On the other hand, our protocols are applicable to wider ranges of ϵ; see Figs. 8 (d–f).

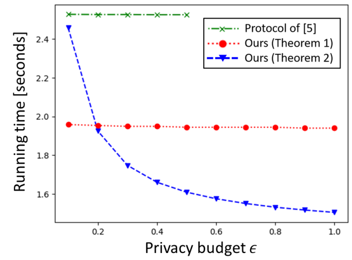
As a comparison between our protocols, we note that the first protocol (Theorem 1) is faster than the second one (Theorem 2) if ϵ gets close to 0 in Figs. 8 (a–c). This is because the communication complexity increases as $\epsilon \to 0$ and then it becomes a dominant factor of the running time. An advantage of the second protocol is that under the strong constraint of utility, it is even applicable to ϵ such that the first one is not; see Figs. 8 (d–f). It comes from the fact that it has the zero probability of failure and as a result, satisfies higher utility with the same communication cost.



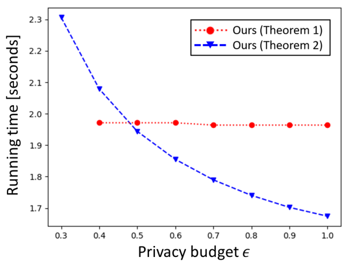
(a) $\alpha = 50$, $\beta = 0.2$ and $\delta = 2^{-100}$



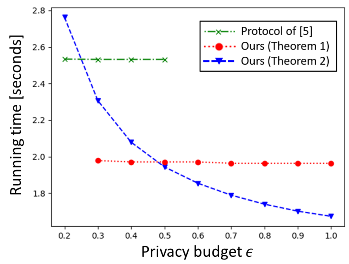
(b) $\alpha = 50$, $\beta = 0.2$ and $\delta = 2^{-80}$



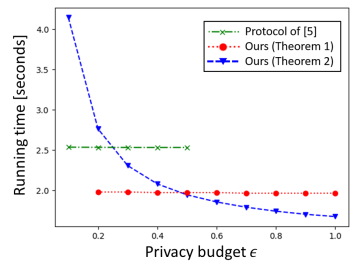
(c) $\alpha = 50$, $\beta = 0.2$ and $\delta = 2^{-60}$



(d) $\alpha = 10$, $\beta = 0.2$ and $\delta = 2^{-100}$



(e) $\alpha = 20$, $\beta = 0.2$ and $\delta = 2^{-100}$



(f) $\alpha = 30$, $\beta = 0.2$ and $\delta = 2^{-100}$

Fig. 8: Estimation of running times to generate a discrete Laplace noise

TABLE 2: Comparison of protocols generating noise from $\mathsf{NBin}(N, M)$

Reference	Communication	Round	Differential privacy	(α, β)-Utility
[5]	$O(N)$	2	(ϵ, δ)-DP for $\epsilon \geq \epsilon(\delta, N, M, \Delta)$	$\alpha = \sqrt{(N/2M^2)\ln(2/\beta)}$
[5]	$O(N/n)$	1	(ϵ, δ)-DP for $\epsilon \geq \epsilon(\delta, N(1-t/n), M, \Delta)$	$\alpha = \sqrt{(N/2M^2)\ln(2/\beta)}$
Theorem 3	$O(\lambda n^{t+1})$ (in the setup)	1 (in the setup)	Computational (ϵ, δ)-DP for $\epsilon \geq \epsilon(\delta, N/\binom{n}{t}, M, \Delta)$	$\alpha = \sqrt{(N/2M^2)\ln(2/(\beta - \delta_\psi))}$

6.2 The Binomial Distribution

We compare our protocol in Section 5 with the ones in [5]. The first protocol of [5] generates shares of N random bits using Σ_{Bit} and then locally computes a share of $Z \sim \mathsf{NBin}(N, M)$. The second protocol of [5] lets each party i share N/n (local) random bits among the other parties and then locally compute a share of Z. Since tN/n out of the N bits are revealed to the adversary, the latter is (ϵ, δ)-differentially private only for $\epsilon \geq \epsilon(\delta, N(1-t/n), M, \Delta)$.

6.2.1 *Communication Complexity*

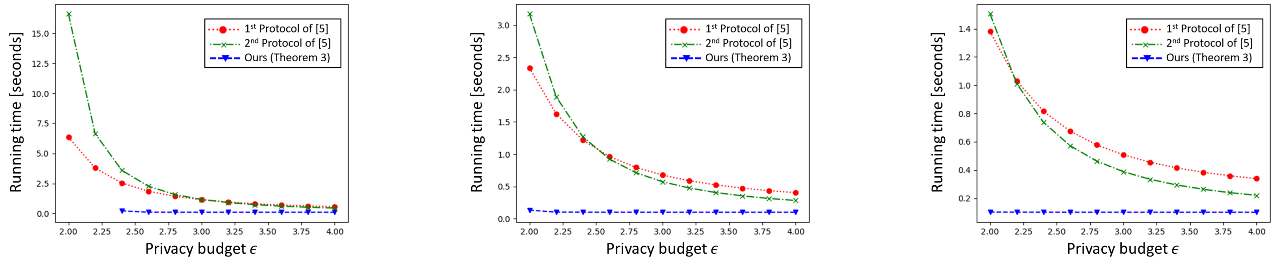
At the cost of predistributing keys for a pseudorandom function with output length ℓ, our protocol non-interactively generates shares of $Z \sim \mathsf{NBin}(N, M)$ by using the keys N/ℓ times (if overflow does not occur). The communication cost to distribute the keys is $\lambda(n-t)\binom{n}{t} = O(\lambda n^{t+1})$, independent of N or M. Furthermore, the communication cost in the setup can be amortized by reusing the keys. However, both of the protocols [5] require communication complexity proportional to N when generating $Z \sim \mathsf{NBin}(N, M)$. Approximately, N grows proportionally to $(\Delta/\epsilon)^2$ in view of the condition (1). In addition, when a fixed-point data type with resolution 2^{-k} is dealt with, N should satisfy the condition (1) for $\Delta = \max\{|g(\boldsymbol{x}) - g(\boldsymbol{y})|2^k : \boldsymbol{x}, \boldsymbol{y} \text{ are neighboring}\}$, which means that N grows exponentially in the length of the fractional part of the data type.

6.2.2 *Utility*

To achieve the same level of differential privacy ϵ and δ, the error bound α of our protocol is $\binom{n}{t}^{1/2} = O(n^{t/2})$ times larger than [5]. Nevertheless, we should also consider the client-server model for practical applications. Our protocol is available and even suitable for this model since n corresponds to the number of servers and is typically small, e.g., $n = 3$.

6.2.3 *Concrete Comparison*

We estimate the running time required by the protocols of [5] and ours (Theorem 3). Consider the same setting as Section 6.1.4. That is, we set $n = 3$, $t = 1$, $\Delta = 1$ and $q = 2^{61} - 1$. We again assume the WAN setting, and set the network speed to $B = 100\,\text{Mbits}/\sec$ and the latency to $0.1\sec$. We use the pseudorandom function based on AES with $\lambda = 128$ and $s = \ell = 128$, assuming $\delta_\psi = 0$. The local computation is considered to be dominated by the execution of AES. We thus calculate the running time as $(\#\,\text{comm. bits})/B + (\#\,\text{rounds}) \times (\text{latency})$ for the protocols [5] and as $(\#\,\text{comm. bits})/B + (\#\,\text{rounds}) \times (\text{latency}) + 100 t_{\text{AES}} N/\ell$ for our protocol, where t_{AES} is the running time of one AES execution. We use the value $t_{\text{AES}} = 3.5\,\text{n}\sec$ [44]. Amortized times to generate 100 binomial noises are shown in Fig. 9, where the privacy budget ϵ ranges from 2 to 4, for $\alpha = 100, 150$ and 200, with $\beta = 0.2$ and $\delta = 2^{-20}$. Our protocol only communicates $\lambda(n-t)\binom{n}{t} = 768$ bits independent of ϵ or δ. We figure



(a) $\alpha = 100$, $\beta = 0.2$ and $\delta = 2^{-20}$ **(b)** $\alpha = 150$, $\beta = 0.2$ and $\delta = 2^{-20}$ **(c)** $\alpha = 200$, $\beta = 0.2$ and $\delta = 2^{-20}$

Fig. 9: Estimation of running times to generate 100 binomial noises

out that the running times of ours lie between $0.1\,\mathrm{sec}$ and $0.22\,\mathrm{sec}$ for all privacy budgets. On the other hand, those of [5] become higher if ϵ gets close to 0, i.e., higher differential privacy is required. Our protocol is up to around 21 times faster for $\alpha = 100$, 23 times for $\alpha = 150$ and 14 times for $\alpha = 200$.

7 Extension to Active Security

We demonstrate that MPC protocols in our framework in Section 3 can achieve active security if actively secure protocols for evaluating deterministic functions and for jointly generating uniformly random elements are given. In our protocols in Sections 4 and 5, the task of generating uniformly random elements is equivalent to giving parties random shares of Shamir and replicated secret sharing schemes, respectively. Therefore, our protocols can provide differential privacy even in the presence of active adversaries if the corruption threshold t is less than $n/3$.

To be precise, we first extend Definition 1 to the active setting. We define the ideal process and the real process. Consider an MPC protocol Π associated with an n-party randomized functionality $\mathcal{F}$.

Ideal Process: This process is defined with respect to a trusted party. A subset of parties T can be corrupted by a PPT ideal process adversary $\mathcal{B}$. The process proceeds in the following steps:

1) *Inputs*: The i-th party obtains an input x_i.
2) *Sending inputs to the trusted party*: An honest party $i \notin T$ always sends x_i to the trusted party. A malicious party $i \in T$ may, depending on x_i, either abort or send some x'_i to the trusted party.
3) *Trusted party answers i-th party*: Suppose that the trusted party receives inputs x'_i from the i-th party. It sends the i-th output y_i to the i-th party, where $\mathcal{F}(x'_1, \ldots, x'_n) = (y_1, \ldots, y_n)$.
4) *Outputs*: If the i-th party is honest, it outputs y_i. The adversary $\mathcal{B}$ outputs an arbitrary (polynomial-time computable) function of $\boldsymbol{x}_T$ and the message it has obtained from the trusted party.

We define $\mathsf{Ideal}^{\mathcal{F}}_{\mathcal{B}}(\boldsymbol{x})$ as the distribution defined over the view of $\mathcal{B}$.

Real Process: The i-th party receives the input x_i. All the parties then execute the protocol Π. A subset of parties T is controlled by an adversary $\mathcal{A}$, who can deviate arbitrarily from the rules of the protocol. We define $\mathsf{View}^{\Pi}_{\mathcal{A}}(\boldsymbol{x})$ as the distribution over the view of $\mathcal{A}$.

We denote by $\mathcal{F}_g$ a functionality of computing a deterministic function g, that is, $\mathcal{F}_g$ outputs $g(\boldsymbol{x})$ if it receives $\boldsymbol{x}$ from the parties.

Definition 2. We call Π a $(t; \epsilon, \delta)$-differentially private protocol for computing g with (α, β)-utility in the presence of active adversaries if for any adversary $\mathcal{A}$ in the real process corrupting a subset of t parties T, there exists a PPT adversary $\mathcal{B}$ in the ideal process for $\mathcal{F}_g$ such that:

- t-Privacy: for any input $\boldsymbol{x}$, the distributions of $\mathsf{View}^{\Pi}_{\mathcal{A}}(\boldsymbol{x})$ and $\mathsf{Ideal}^{\mathcal{F}_g}_{\mathcal{B}}(\boldsymbol{x})$ are identical.
- (ϵ, δ)-Differential privacy: for any pair of T-neighboring inputs $\boldsymbol{x}$ and $\boldsymbol{y}$, the distributions of $\mathsf{View}^{\Pi}_{\mathcal{A}}(\boldsymbol{x})$ and $\mathsf{View}^{\Pi}_{\mathcal{A}}(\boldsymbol{y})$ are (ϵ, δ)-DP close.
- (α, β)-Utility: for any input $\boldsymbol{x}$ and $i \notin T$, it holds that
$$\Pr\left[|\mathsf{Output}^{\Pi}_{\mathcal{A},i}(\boldsymbol{x}) - g(\boldsymbol{x}^{\mathcal{B}})| \leq \alpha\right] \geq 1 - \beta,$$
where $\mathsf{Output}^{\Pi}_{\mathcal{A},i}(\boldsymbol{x})$ is a part of the output the i-th party receives at the end of the real process and $\boldsymbol{x}^{\mathcal{B}}$ is a tuple of inputs $(\boldsymbol{x}_{[n]\setminus T}, \boldsymbol{x}'_T)$ submitted to the trusted party in the ideal process.

As in Proposition 1, let $g : D^n \to \mathbb{Z}_q$ be a deterministic function to compute and $h : U \to \mathbb{Z}_q$ be a noise generator function. Let t be a corruption threshold with $t < n/3$. Then, there is a protocol Π_g (resp. Π_h) which takes $\boldsymbol{x} \in D^n$ (resp. $\boldsymbol{s} \in U$) as input and t-securely computes $(\llbracket g(\boldsymbol{x}) \rrbracket_i)_{i \in [n]}$ (resp. $(\llbracket h(\boldsymbol{s}) \rrbracket_i)_{i \in [n]}$) in the presence of active adversaries. Suppose that we are also given an actively secure protocol Σ_{Ran} realizing a functionality $\mathcal{F}_{\mathrm{Ran}}$, which takes no input, samples $\boldsymbol{s} = (s_i)_{i \in [n]} \sim \mathsf{Uni}(U)$, and gives s_i to the i-th party. Also, suppose that, for any subset T of size t, any pair of T-neighboring vectors $\boldsymbol{x}, \boldsymbol{y} \in D^n$, and any $\boldsymbol{a} \in U_T$, two distributions $(\boldsymbol{x}_T, \boldsymbol{s}_T, g(\boldsymbol{x}) + h(\boldsymbol{s}))$, $(\boldsymbol{y}_T, \boldsymbol{s}_T, g(\boldsymbol{y}) + h(\boldsymbol{s}))$ conditioned on $\boldsymbol{s}_T = \boldsymbol{a}$ are (ϵ, δ)-DP close, where the randomness is $\boldsymbol{s} \sim \mathsf{Uni}(U)$. Finally, assume that $\Pr_{\boldsymbol{s} \sim \mathsf{Uni}(U)}[|h(\boldsymbol{s})| \leq \alpha] \geq 1 - \beta$ for α and β. Now, we construct a protocol Π based on Π_g, Π_h, and Σ_{Ran} (Fig. 10).

Proposition 5. Using the above notations, the protocol Π described in Fig. 10 is a $(t; \epsilon, \delta)$-differentially private protocol for computing g with (α, β)-utility in the presence of active adversaries.

Input. $\boldsymbol{x} \in D^n$.
Output. The i-th party receives (s_i, z), where $\boldsymbol{s} = (s_i)_{i\in[n]} \sim \mathsf{Uni}(U)$ and $z = g(\boldsymbol{x}) + h(\boldsymbol{s})$.
Protocol.
1) $[\![g(\boldsymbol{x})]\!] = \Pi_g(\boldsymbol{x})$.
2) $\boldsymbol{s} = \Sigma_{\text{Ran}}()$.
3) $[\![h(\boldsymbol{s})]\!] = \Pi_h(\boldsymbol{s})$.
4) $[\![z]\!] = [\![g(\boldsymbol{x})]\!] + [\![h(\boldsymbol{s})]\!]$.
5) The i-th party receives the shares $[\![z]\!]$ and outputs (s_i, z).

Fig. 10: A $(t; \epsilon, \delta)$-differentially private protocol for computing g in the presence of active adversaries

Proof: Define $\mathcal{F}$ as a functionality which takes $\boldsymbol{x} \in D^n$ as input, samples $\boldsymbol{s} \sim \mathsf{Uni}(U)$, and gives $(s_i, f(\boldsymbol{x}; \boldsymbol{s}))$ to the i-th party, where $f(\boldsymbol{x}; \boldsymbol{s}) := g(\boldsymbol{x}) + h(\boldsymbol{s})$. Then, the protocol Π t-securely realizes $\mathcal{F}$. Indeed, Π simply invokes actively secure protocols Π_g, Π_h, and Σ_{Ran} sequentially and opens z to every party. We apply the composition theorem [31] and use the robustness of Shamir secret sharing scheme since $t < n/3$.

Then, Π satisfies $(t; \epsilon, \delta)$-differential privacy for computing g with (α, β)-utility. To see this, let $\mathcal{A}$ be an adversary in the real process. From the security of Π for $\mathcal{F}$, we have an adversary $\mathcal{B}'$ in the ideal process for $\mathcal{F}$ corresponding to $\mathcal{A}$. In particular, there exists a simulator Sim' such that $\mathsf{View}^{\Pi}_{\mathcal{A}}(\boldsymbol{x})$ is identical to $\mathsf{Sim}'(\boldsymbol{x}_T, \mathcal{F}(\boldsymbol{x}^{\mathcal{B}'})_T)$ for any $\boldsymbol{x}$, where $\boldsymbol{x}^{\mathcal{B}'}$ is a tuple of inputs $(\boldsymbol{x}_{[n]\setminus T}, \boldsymbol{x}'_T)$ submitted by $\mathcal{B}'$ and $\mathcal{F}(\boldsymbol{x}^{\mathcal{B}'})_T = (s_i, f(\boldsymbol{x}^{\mathcal{B}'}; \boldsymbol{s}))_{i\in T}$ for $\boldsymbol{s} \sim \mathsf{Uni}(U)$. The t-privacy of Π then follows since we can obtain an adversary $\mathcal{B}$ in the ideal process for $\mathcal{F}_g$ from $\mathcal{B}'$ and from a simulator Sim which on input $\boldsymbol{x}_T$ and $g(\boldsymbol{x}^{\mathcal{B}'})$, locally samples $\boldsymbol{s} \sim \mathsf{Uni}(U)$ and runs Sim' on $(\boldsymbol{x}_T, (s_i, g(\boldsymbol{x}^{\mathcal{B}'}) + h(\boldsymbol{s}))_{i\in T})$. For differential privacy, observe that if $\boldsymbol{x}$ and $\boldsymbol{y}$ are T-neighboring, then so are $\boldsymbol{x}^{\mathcal{B}'}$ and $\boldsymbol{y}^{\mathcal{B}'}$ since $\mathcal{B}'$ substitutes corrupted parties' inputs depending only on $\boldsymbol{x}_T = \boldsymbol{y}_T$. Thus, the distributions $(\boldsymbol{x}_T, \boldsymbol{s}_T, f(\boldsymbol{x}^{\mathcal{B}'}; \boldsymbol{s}))$ and $(\boldsymbol{y}_T, \boldsymbol{s}_T, f(\boldsymbol{y}^{\mathcal{B}'}; \boldsymbol{s}))$ are (ϵ, δ)-DP close if $\boldsymbol{s} \sim \mathsf{Uni}(U)$. The existence of Sim' and the post-processing property of differential privacy imply that $\mathsf{View}^{\Pi}_{\mathcal{A}}(\boldsymbol{x})$ and $\mathsf{View}^{\Pi}_{\mathcal{A}}(\boldsymbol{y})$ are (ϵ, δ)-DP close. The (α, β)-utility of Π can be derived from that of $g(\cdot) + h(\boldsymbol{s})$, $\boldsymbol{s} \sim \mathsf{Uni}(U)$ and from the fact that $\mathsf{Output}^{\Pi}_{\mathcal{A},i}(\boldsymbol{x})$ is guaranteed to contain $f(\boldsymbol{x}^{\mathcal{B}'}; \boldsymbol{s})$ for $\boldsymbol{s} \sim \mathsf{Uni}(U)$ by the security of Π. □

We actually have the protocols Π_g, Π_h, and Σ_{Ran} (e.g., [18], [19], [36]) if g and h are represented as arithmetic circuits and U is the set of all possible shares of Shamir or replicated secret sharing schemes. This is the case for our protocols in Sections 4 and 5 and hence they can provide differential privacy even in the presence of active adversaries.

For concrete efficiency, we estimate the running times for our actively secure protocols to generate discrete Laplace and binomial noises in the same way as Section 6. For the protocols in Theorems 1 and 2, we calculate additional communication bits and rounds of interaction based on the actively secure MPC protocol of [45]. In Fig. 11 (a), we compare the running times of our actively secure protocols to those of the passive ones in the same setting as Fig. 8 (f) except that we set $n = 4 > 3t$. As shown, the running time of the first protocol is at most four times slower and that of the second one is at most 16 times slower. We also estimate the running time taking into account the local computation based on the experimental results [24, Table 5] (in the active-security-with-abort model). They show that it takes at most $28\,\text{sec}$ to evaluate a circuit of depth 20 and size 10^6 for a small number of parties $n \le 5$. Since the circuits considered in Theorems 1 and 2 belong to the above set of circuits for the parameters in Section 6.1.4, we conclude that our protocols can generate noises in the presence of active adversaries for a reasonable time.

For the one in Theorem 3, notice that an additional cost to achieve active security is only that we have to robustly generate a random share for replicated secret sharing (all the other computations are locally done). Based on the sharing protocol of [36], the running time of the actively secure protocol is estimated as at most four times slower than the passive one. It can be empirically demonstrated by Fig. 11 (b) in the same setting as Fig. 9 (c) except that we set $n = 4 > 3t$.

8 Conclusion

In this paper, we propose three efficient MPC protocols for generating shares of noise drawn from certain distributions capable of providing differential privacy. Our protocols for the discrete Laplace distribution improves the round complexity of the previous one [5]. Our third protocol enables parties to non-interactively compute shares of noise drawn from the binomial distribution by predistributing keys for a pseudorandom function. It reduces the communication complexity of [5] while it loses utility to some extent. Our protocols can be extended so that they provide differential privacy even in the presence of active adversaries.

Acknowledgments

This research was partially supported by JSPS KAKENHI Grant Numbers JP20J20797 and JP19K22838, JST CREST Grant Numbers JPMJCR14D6 and JPMJCR19F6, Japan, and the Ministry of Internal Affairs and Communications SCOPE Grant Number 182103105.

References

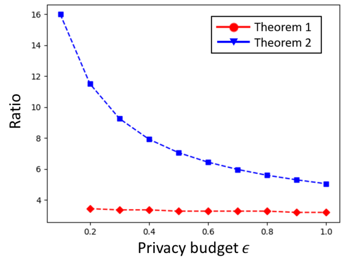
[1] D. Bogdanov, R. Talviste, and J. Willemson, "Deploying secure multi-party computation for financial data analysis," in *Int. Conf. Fin. Cryptogr. Data Security*, 2012, pp. 57–64.

[2] E. Kimura, K. Hamada, R. Kikuchi, K. Chida, K. Okamoto, S. Manabe, T. Kuroda, Y. Matsumura, T. Takeda, and N. Mihara, "Evaluation of secure computation in a distributed healthcare setting," *Stud. Health Techn. Inform.*, vol. 228, pp. 152–156, 2016.

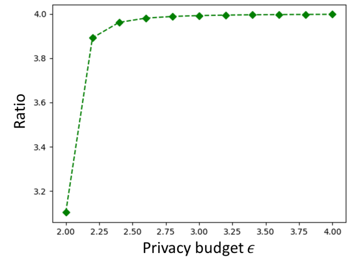
[3] J. A. Calandrino, A. Kilzer, A. Narayanan, E. W. Felten, and V. Shmatikov, ""You might also like:" privacy risks of collaborative filtering," in *2011 IEEE Symp. Security Privacy*, 2011, pp. 231–246.

[4] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *2017 IEEE Symp. Security Privacy*, 2017, pp. 3–18.

[5] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, "Our data, ourselves: Privacy via distributed noise generation," in *Adv. Cryptol. EUROCRYPT 2006*, 2006, pp. 486–503.



(a) Generation of a Laplace noise. ($\alpha = 30$, $\beta = 0.2$ and $\delta = 2^{-60}$)



(b) Generation of 100 binomial noises. ($\alpha = 200$, $\beta = 0.2$ and $\delta = 2^{-20}$)

Fig. 11: Ratio between running times of our active and passive protocols

[6] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory Cryptogr.*, 2006, pp. 265–284.

[7] N. Agarwal, A. T. Suresh, F. X. X. Yu, S. Kumar, and B. McMahan, "cpSGD: Communication-efficient and differentially-private distributed SGD," in *Adv. Neural Inf. Process. Syst.*, 2018, pp. 7564–7575.

[8] A. Beimel, K. Nissim, and E. Omri, "Distributed private data analysis: Simultaneously solving how and what," in *Adv. Cryptol. CRYPTO 2008*, 2008, pp. 451–468.

[9] G. Wu, Y. He, J. Wu, and X. Xia, "Inherit differential privacy in distributed setting: Multiparty randomized function computation," in *2016 IEEE Trustcom/BigDataSE/ISPA*, 2016, pp. 921–928.

[10] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith, "What can we learn privately?" *SIAM J. Comput.*, vol. 40, no. 3, pp. 793–826, 2011.

[11] E. Shi, T.-H. Chan, E. Rieffel, R. Chow, and D. Song, "Privacy-preserving aggregation of time-series data," in *NDSS*, vol. 2, 2011.

[12] R. Bassily and A. Smith, "Local, private, efficient protocols for succinct histograms," in *Proc. 47th Annu. ACM Symp.Theory Comput.*, ser. STOC '15, 2015, pp. 127–135.

[13] T. Wang, J. Blocki, N. Li, and S. Jha, "Locally differentially private protocols for frequency estimation," in *Proc. 26th USENIX Conf. Security Symp.*, ser. SEC'17, 2017, pp. 729–745.

[14] B. Balle, J. Bell, A. Gascón, and K. Nissim, "The privacy blanket of the shuffle model," in *Adv. Cryptol. CRYPTO 2019*, 2019, pp. 638–667.

[15] A. Cheu, A. Smith, J. Ullman, D. Zeber, and M. Zhilyaev, "Distributed differential privacy via shuffling," in *Adv. Cryptol. EUROCRYPT 2019*, 2019, pp. 375–403.

[16] U. Erlingsson, V. Feldman, I. Mironov, A. Raghunathan, K. Talwar, and A. Thakurta, "Amplification by shuffling: From local to central differential privacy via anonymity," in *Proc. 30th Annual ACM-SIAM Symp. Discret. Algo.*, ser. SODA '19, 2019, pp. 2468–2479.

[17] F. Eigner, A. Kate, M. Maffei, F. Pampaloni, and I. Pryvalov, "Differentially private data aggregation with optimal utility," in *Proc. 30th Annual Comput. Security Appl. Conf.*, 2014, pp. 316–325.

[18] I. Damgård and J. B. Nielsen, "Scalable and unconditionally secure multiparty computation," in *Adv. Cryptol. CRYPTO 2007*, 2007, pp. 572–590.

[19] V. Goyal, Y. Liu, and Y. Song, "Communication-efficient unconditional MPC with guaranteed output delivery," in *Adv. Cryptol. CRYPTO 2019*, 2019, pp. 85–114.

[20] R. Stanojevic, M. Nabeel, and T. Yu, "Distributed cardinality estimation of set operations with differential privacy," in *2017 IEEE Symp. Privacy-Aware Comput.*, 2017, pp. 37–48.

[21] R. Cramer, I. Damgård, and Y. Ishai, "Share conversion, pseudorandom secret-sharing and applications to secure computation," in *Theory Cryptogr.*, 2005, pp. 342–362.

[22] I. Mironov, O. Pandey, O. Reingold, and S. Vadhan, "Computational differential privacy," in *Adv. Cryptol. CRYPTO 2009*, 2009, pp. 126–142.

[23] B. Anandan and C. Clifton, "Laplace noise generation for two-party computational differential privacy," in *2015 13th Annual Conf. Privacy Security Trust*, 2015, pp. 54–61.

[24] K. Chida, D. Genkin, K. Hamada, D. Ikarashi, R. Kikuchi, Y. Lindell, and A. Nof, "Fast large-scale honest-majority MPC for malicious adversaries," in *Adv. Cryptol. CRYPTO 2018*, 2018, pp. 34–64.

[25] M. Aliasgari, M. Blanton, Y. Zhang, and A. Steele, "Secure computation on floating point numbers," in *NDSS*, 2013.

[26] I. Mironov, "On significance of the least significant bits for differential privacy," in *Proc. 2012 ACM Conf. Comput. Commun. Security*, 2012, pp. 650–661.

[27] I. Gazeau, D. Miller, and C. Palamidessi, "Preserving differential privacy under finite-precision semantics," *Theoretic. Comput. Sci.*, vol. 655, pp. 92–108, 2016.

[28] S. Wagh, X. He, A. Machanavajjhala, and P. Mittal, "Dp-cryptography: Marrying differential privacy and cryptography in emerging applications," *Commun. ACM*, vol. 64, no. 2, pp. 84–93, 2021.

[29] J. Champion, a. shelat, and J. Ullman, "Securely sampling biased coins with applications to differential privacy," in *Proc. 2019 ACM SIGSAC Conf. Comput. Commun. Security*, 2019, pp. 603–614.

[30] R. Eriguchi, A. Ichikawa, N. Kunihiro, and K. Nuida, "Efficient noise generation to achieve differential privacy with applications to secure multiparty computation," in *Financial Cryptography and Data Security*, 2021, pp. 271–290.

[31] O. Goldreich, *Foundations of cryptography: volume 2, basic applications*. Cambridge University Press, 2009.

[32] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[33] M. Ben-Or, S. Goldwasser, and A. Wigderson, "Completeness theorems for non-cryptographic fault-tolerant distributed computation," in *Proc. 20th Annual ACM Symp. Theory Comput.*, 1988, pp. 1–10.

[34] O. Goldreich, S. Goldwasser, and S. Micali, "How to construct random functions," *J. ACM*, vol. 33, no. 4, pp. 792–807, 1986.

[35] A. B. Lewko and B. Waters, "Efficient pseudorandom functions from the decisional linear assumption and weaker variants," in *Proc. 16th ACM Conf. Comput. Commun. Security*, ser. CCS '09, 2009, pp. 112–120.

[36] M. Hirt and D. Tschudi, "Efficient general-adversary multi-party computation," in *Adv. Cryptol. ASIACRYPT 2013*, 2013, pp. 181–200.

[37] T. Nishide and K. Ohta, "Multiparty computation for interval, equality, and comparison without bit-decomposition protocol," in *Public Key Cryptogr. 2007*, 2007, pp. 343–360.

[38] I. Damgård, M. Fitzi, E. Kiltz, J. B. Nielsen, and T. Toft, "Unconditionally secure constant-rounds multi-party computation for equality, comparison, bits and exponentiation," in *Theory Cryptogr.*, 2006, pp. 285–304.

[39] O. Catrina and S. de Hoogh, "Improved primitives for secure multiparty integer computation," in *Security Cryptogr. Netw.*, 2010, pp. 182–199.

[40] S. Inusah and T. J. Kozubowski, "A discrete analogue of the Laplace distribution," *J. Stat. Planning Infer.*, vol. 136, no. 3, pp. 1090–1102, 2006.

[41] A. Ghosh, T. Roughgarden, and M. Sundararajan, "Universally utility-maximizing privacy mechanisms," *SIAM J. Comput.*, vol. 41, no. 6, pp. 1673–1693, 2012.

[42] K. Park, H. Park, W.-C. Jeun, and S. Ha, "Boolean circuit programming: A new paradigm to design parallel algorithms," *J. Discret. Algo.*, vol. 7, no. 2, pp. 267–277, 2009.

[43] J. Böhler and F. Kerschbaum, "Secure multi-party computation of differentially private heavy hitters," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '21, 2021, pp. 2361–2377.

[44] M. Chase, E. Ghosh, and O. Poburinnaya, "Secret-shared shuffle," in *Adv. Cryptol. ASIACRYPT 2020*, 2020, pp. 342–372.
[45] Z. Beerliová-Trubíniová and M. Hirt, "Perfectly-secure MPC with linear communication complexity," in *Theory Cryptogr.*, 2008, pp. 213–230.