Securing Dynamic Data: A Primer on Differentially Private Data Structures

Monika Henzinger
Institute of Science and Technology, Klosterneuburg, Austria

Roodabeh Safavi
Institute of Science and Technology, Klosterneuburg, Austria

Abstract

We give an introduction into differential privacy in the dynamic setting, called the continual observation setting.

2012 ACM Subject Classification Security and privacy; Theory of computation

Keywords and phrases Differential privacy, continual observation

Digital Object Identifier 10.4230/LIPIcs.ESA.2025.2

Category Invited Talk

1 Introduction

In 2006, Netflix released an anonymized dataset of movie ratings for a public competition. Although names were removed, it was later shown that users could be re-identified by linking the data with publicly available IMDb ratings [39]. This so-called (privacy) attack demonstrated the failure of traditional anonymization techniques, especially when different anonymized datasets are combined. It also highlighted the need for stronger privacy definitions that provide rigorous guarantees even when multiple analyses (or synthetic datasets) based on the same individual's secret data are published.

This motivated the definition of differential privacy (DP), a privacy requirement for mechanisms introduced by Dwork, McSherry, Nissim, and Smith [17]. A differentially private mechanism[1] adds randomness to the output in such a way that, for any two "neighboring" datasets, the resulting output distributions are "close". Formally, the notion of neighboring datasets is defined as a binary relation on the family of datasets, and closeness of output distributions is defined by the following definition of indistinguishability:

▶ **Definition 1** ((ϵ, δ)-indistinguishability). *For $\epsilon \geq 0$ and $0 \leq \delta \leq 1$, two random variables X_0 and X_1 over a domain $\mathcal{Y}$ are (ϵ, δ)-indistinguishable if, for every subset $Y \subseteq \mathcal{Y}$,*

$$Pr[X_0 \in Y] \leq e^{\epsilon} Pr[X_1 \in Y] + \delta.$$

X_0 and X_1 are ϵ-indistinguishable if $\delta = 0$.

Two datasets are typically considered to be neighboring if they differ by the presence or absence of a single data record. With this definition of neighboring, after observing the output of a differentially private mechanism, an attacker cannot confidently determine

[1] In theory, a common way to achieve differential privacy is by adding noise drawn from continuous distributions, such as the Gaussian distribution, when computing data analysis. However, exact sampling from continuous distributions is computationally infeasible. For this reason, the differential privacy literature typically avoids the term "differentially private algorithm" and uses the term "mechanism" instead. In practice, discrete approximations of these continuous distributions – implemented with finite precision – are used instead.

whether any specific data record was included in the input dataset. Let $\mathcal{X}$ denote the family of data records, and $\mathrm{MS}(\mathcal{X})$ denote the set of all multisets over $\mathcal{X}$. A differentially private mechanism is formally defined as follows:

▶ **Definition 2** ((ϵ, δ)-Differential Privacy [17]). *For $\epsilon \geq 0$ and $0 \leq \delta \leq 1$, a randomized mechanism $\mathcal{M} : MS(\mathcal{X}) \to \mathcal{Y}$ is (ϵ, δ)-differentially private (or (ϵ, δ)-DP for short) if, for every two neighboring datasets $D_0, D_1 \in MS(\mathcal{X})$, the random variables $\mathcal{M}(D_0)$ and $\mathcal{M}(D_1)$ are (ϵ, δ)-indistinguishable. $\mathcal{M}$ is ϵ-differentially private (or ϵ-DP for short) if $\delta = 0$.*

Note that there exist problems that can be solved efficiently and with high accuracy in the non-private setting, but for which no differentially private solution exists. The interior point problem is an instance of this: Suppose the data universe $\mathcal{X}$ is totally ordered and datasets are multisets of size $n \in N$ over $\mathcal{X}$. An interior point mechanism takes a dataset D as input and its goal is to return an element $y \in \mathcal{X}$ such that y is an interior point in D, i.e., $\min_{x \in D} x \leq y \leq \max_{x \in D} x$. An interior point mechanism is said to be accurate if for every input dataset, it returns an interior point of the dataset with probability at least 3/4. Differential privacy for this mechanism is defined with respect to neighboring datasets that differ in exactly one element. Bun, Nissim, Stemmer, and Vadhan [6] showed that for any $0 < \epsilon < 1/4$ and $\delta < 1/(50n^2)$, there exists no accurate interior point mechanism that satisfies (ϵ, δ)-DP.

Fundamental properties of differential privacy. Differential privacy is carefully designed to satisfy the following two crucial properties: (i) It is preserved under post-processing, and (ii) executing multiple differentially private mechanisms on the same private dataset remains differentially private, with the privacy parameters determined by those of the executed mechanisms. The first property ensures that any computation applied to the output of a differentially private mechanism does not incur additional privacy loss. For example, a privatized statistic released by a hospital can be safely reused in multiple research studies to derive new results, without further compromising the privacy of the patients. The second property, known as *composition*, provides a bound on the total privacy loss when an individual's data is used in multiple differentially private analyses. The goal of this property is to prevent attacks similar to the Netflix attack discussed before.

In *post-processing*, a subsequent mechanism that does *not* have access to the sensitive dataset uses the output of a DP mechanism as input. The so-called *post-processing lemma* states that this does not affect the differential privacy properties; informally speaking, no more privacy is lost.

▶ **Lemma 3** (Post-Processing [20]). *Let $\mathcal{M} : MS(\mathcal{X}) \to \mathcal{Y}$ be an (ϵ, δ)-DP mechanism, and $f : \mathcal{Y} \to \mathcal{Z}$ be a function. Then, the mechanism $f(\mathcal{M})$, which applies $\mathcal{M}$ to the input and then applies f to its output, is also (ϵ, δ)-DP.*

A different situation arises if the second mechanism has access to the sensitive data as well. In this case, for every output of the first mechanism, the second mechanism must also satisfy differentially private. The following two so-called *composition theorems* analyze the total privacy loss for the combination of the output of multiple mechanisms.

▶ **Theorem 4** (Basic Composition [20]). *Let $\mathcal{M}_1 : MS(\mathcal{X}) \to \mathcal{Y}_1$ and $\mathcal{M}_2 : MS(\mathcal{X}) \to \mathcal{Y}_2$ be mechanisms that satisfy (ϵ_1, δ_1)-DP and (ϵ_2, δ_2)-DP, respectively. Then, the mechanism $\mathcal{M}$, defined as $\mathcal{M}(D) = (\mathcal{M}_1(D), \mathcal{M}_2(D))$ for each $D \in MS(\mathcal{X})$, is $(\epsilon_1 + \epsilon_2, \delta_1 + \delta_2)$-DP.*

By applying basic composition inductively, we obtain that the composition of $k \in \mathbb{N}$ mechanisms, each satisfying (ϵ_0, δ_0)-DP, ensures $(k\epsilon_0, k\delta_0)$-DP. The next theorem, known as *advanced composition theorem*, shows that when ϵ_0 is small, composition also satisfies $(\sqrt{k}\sqrt{\log(1/\delta)}\epsilon_0, \delta + k\delta_0)$-DP for every $\delta \in (0, 1]$, where the first privacy parameter scales with $\sqrt{k}$ rather than k. However, unlike basic composition, advanced composition introduces a nonzero δ even if each individual mechanism is purely differentially private (i.e., $\delta_0 = 0$). This additive term can be interpreted as a small probability of privacy failure. Thus, in applications where any such failure is unacceptable and composition must satisfy pure differential privacy, basic composition must be applied.

▶ **Theorem 5** (Advanced Composition [21]). *Let $k \in \mathbb{N}$, and for each $i \in [k]$, let $\mathcal{M}_i : MS(\mathcal{X}) \to \mathcal{Y}_i$ be an (ϵ_i, δ_i)-DP mechanism. Then, for every $0 \le \delta \le 1$, the mechanism $\mathcal{M}$, defined as $\mathcal{M}(D) = (\mathcal{M}_1(D), \ldots, \mathcal{M}_k(D))$ for each $D \in MS(\mathcal{X})$, is (ϵ, δ)-DP, where*

$$\epsilon = \sum_{i=1}^{k} \epsilon_i \cdot (e^{\epsilon_i} - 1) + \sqrt{2\log(\frac{1}{\delta'}) \sum_{i=1}^{k} \epsilon_i^2} \qquad \text{and} \qquad \delta = \delta' + \sum_{i=1}^{k} \delta_i.$$

The next composition theorem guarantees differential privacy when multiple differentially private analyses are performed on disjoint subsets of a dataset. In this composition, known as the *parallel composition*, the overall privacy loss is determined by the *maximum* privacy loss of the individual mechanisms, rather than the sum. For example, consider multiple elementary schools, each with a distinct set of students. Suppose each school applies an (ϵ, δ)-differentially private mechanism to analyze its own student data. If the government later aggregates the privatized outputs from all schools for policy decisions, the overall process remains (ϵ, δ)-differentially private.

▶ **Definition 6** (Parallel Composition [38]). *Let $k \in \mathbb{N}$, and for each $i \in [k]$, let $\mathcal{M}_i : MS(\mathcal{X}) \to \mathcal{Y}_i$ be an (ϵ_i, δ_i)-DP mechanism with respect to a neighbor relation $\sim_i$. Define $\mathcal{M} : MS(\mathcal{X})^k \to \mathcal{Y}_1 \times \cdots \times \mathcal{Y}_k$ as $\mathcal{M}(D) = (\mathcal{M}_1(D_1), \ldots, \mathcal{M}_k(D_k))$ for each $D = (D_1, \ldots, D_k) \in MS(\mathcal{X})^k$. Two sequences of datasets $D = (D_1, \ldots, D_k) \in MS(\mathcal{X})^k$ and $D' = (D'_1, \ldots, D'_k) \in MS(\mathcal{X})^k$ are considered to be neighboring if and only if there exists $i \in [k]$ such that $D_i \sim_i D'_i$ and $D_j = D'_j$ for all $j \neq i$. Then, the mechanism $\mathcal{M}$ satisfies (ϵ, δ)-DP, where $\epsilon = \max_{1 \le i \le k} \epsilon_i$ and $\delta = \max_{1 \le i \le k} \delta_i$.*

The Laplace Mechanism. One of the most fundamental differentially private mechanisms is the *Laplace Mechanism*. This mechanism computes a "simple" statistic f on the input dataset, adds Laplace noise to it, and outputs the noisy result. Here, "simple" refers to functions with low *sensitivity*, where the sensitivity of a function measures how much its output can differ over neighboring datasets:

▶ **Definition 7** (ℓ_k-Sensitivity). *Let $f : MS(\mathcal{X}) \to \mathbb{R}^d$ be a function mapping a dataset to a real number. For each $k \in \mathbb{N}$, the ℓ_k-*sensitivity of f, *denoted $\Delta_{k,f}$, is defined as*

$$\Delta_{k,f} = \max_{D, D' \in MS(\mathcal{X})} \|f(D) - f(D')\|_k ,$$

where $\|\cdot\|_k$ is the k-norm.

▶ **Definition 8** (Laplace Distribution). *The Laplace distribution with mean 0 and scale parameter b has the probability density function*

$$f_{Lap(b)}(x) = \frac{1}{2b} \exp\left(\frac{-|x|}{b}\right), \quad x \in \mathbb{R}.$$

A random variable following this distribution is denoted by $Y \sim Lap(b)$ or simply $Lap(b)$.

The following Lemma formally defines the Laplace mechanism and states a trade-off between its privacy and accuracy guarantees:

▶ **Lemma 9** (Laplace Mechanism [17]). *Let $f : MS(\mathcal{X}) \to \mathbb{R}$ be a function with ℓ_1-sensitivity $\Delta_{1,f}$. For $\epsilon \geq 0$, the Laplace mechanism $\mathcal{M}^{f,\epsilon}_{Lap}$ takes a dataset $D \in MS(\mathcal{X})$ as input and outputs $f(D) + Lap(\Delta_{1,f}/\epsilon)$. The mechanism $\mathcal{M}^{f,\epsilon}_{Lap}$ satisfies ϵ-DP. Furthermore, for every $D \in MS(\mathcal{U})$ and $t \geq 0$,*

$$\Pr\left[|\mathcal{M}^{f,\epsilon}_{Lap}(D) - f(D)| \geq t \cdot \Delta_{1,f}/\epsilon\right] \leq e^{-t}.$$

We note that the definition of accuracy depends on the application and may be different in each context. Therefore, we do not provide a general definition and instead define accuracy for each concrete application in the subsequent sections.

Differential privacy in the continual and interactive setting. The mechanisms discussed so far are referred to as *non-interactive mechanisms (NIMs)* or *batch* mechanisms, as they take a single input dataset and produce a single output. In the rest of this paper, we define settings where either multiple different queries are asked to the mechanism, resulting in an *interactive mechanism (IM)*, or the dataset is changed through update operation, resulting in a *continual mechanism (CM)*.

Note that these mechanisms are differential privacy analogues of randomized algorithms and data structures in the non-private setting: A non-interactive mechanism corresponds to a *static* (or *batch*) *algorithm*, while an interactive mechanism corresponds to *static data structure*. A continual mechanism allows both queries and updates and corresponds to *dynamic data structures*. While the adaptivity of the sequence of operations is a common concern for the correctness and running time of randomized data structures, in differential privacy, adaptivity is additionally a concern for privacy. Concretely, note that the above definitions and composition theorems only hold for non-interactive mechanisms. In the following sections, we will discuss how to define and analyze the privacy properties of interactive and continual mechanisms.

2 The Continual Setting

In many real-world applications, data is not available in a single static batch but is collected incrementally over time or the dataset is changing over time. In such settings, it is often necessary to update the output continually in order to provide timely and relevant analysis of the dataset. For example, an online movie platform may collect user interactions, such as who watches which movie, and update its recommendation model accordingly. In this scenario, the platform must ensure that the sequence of recommendations over time does not reveal sensitive information, such as whether a user has watched a specific movie. The problem of dynamically receiving data and updating the output while protecting the privacy of each data point is studied in the *continual observation model* of differential privacy [18, 10], which has recently become a popular research topic (see e.g. [8, 9, 26, 32, 33, 2, 1, 16, 31, 4, 23] for CMs for various problems on sequences of numbers, [34, 29, 46] for CMs for set cardinality, [11, 40, 14] for CMs for problems in databases, [7, 30] for CMs for various histogram queries, [42, 25, 26, 41, 22] for CMs for various graph problems, [15] for CMs for clustering problems in the Euclidean space, and [45, 5] for CMs for releasing synthetic data).

In the non-private setting, the goal is to design dynamic algorithms that are accurate, memory-efficient, and time-efficient. In the private setting, however, one must additionally ensure the privacy of the input data, which is always in conflict with accuracy. Nonetheless, with the emergence of new algorithmic techniques, several continual mechanisms have

been developed that provide meaningful trade-offs between privacy and accuracy. These mechanisms have been shown to be practical and are already deployed in real-world systems [24, 3, 12, 13, 47, 48].

Formally, a mechanism $\mathcal{M}$ in the continual observation model is defined by a randomized function $f_{\mathcal{M}}$ that maps a current state and an incoming update to a new state and an output. The state represents the internal memory of the mechanism and may, for example, include the current dataset or relevant summary statistics. For simplicity, we assume the mechanism starts from an initial state corresponding to an empty dataset, though this assumption can be relaxed.

The mechanism $\mathcal{M}$ processes a sequence of update instructions $\sigma = (x_1, x_2, \dots)$ as follows[2]: Starting from the (empty) initial state, at each step t, the mechanism applies $f_{\mathcal{M}}$ to its current state and the incoming update x_t, yielding a new internal state and an output value y_t. $\mathcal{M}$ then returns y_t. We denote the sequence of outputs $(y_1, y_2, \dots)$ generated in response to the update sequence σ by the random variable $\mathcal{M}(\sigma)$. We also refer to such a mechanism as a *continual mechanism*.

Depending on the problem definition, each update x_i can be a data record in $\mathcal{X}$ to be inserted into the dataset or an instruction that arbitrarily modifies the current dataset (e.g., it deletes certain records). For simplicity, in the remainder of this section, we assume that each update x_t is a data record in $\mathcal{X}$ to be inserted in the dataset. We note that all the definitions can naturally be extended to non-empty initial datasets and to general update operations.

Recall that in the non-interactive model, the differential privacy guarantee is defined over pairs of neighboring datasets. In the continual setting, on the other hand, the notion of neighboring is defined over sequences of data records. For example, if $\mathcal{X} = \{0, 1\}$, we can define two sequences of data records $\sigma = (x_1, \dots, x_T)$ and $\sigma' = (x'_1, \dots, x'_T)$ of equal length $T \in \mathbb{N}$ to be neighboring if they differ at exactly one position, i.e., there exists $i \in [T]$ such that $x_j = x'_j$ for all $j \neq i$. Using this definition of neighboring in the definition of DP leads to so-called *event-level privacy*.

▶ **Definition 10** ((ϵ, δ)-Differential Privacy for Continual Mechanisms [18, 10])**.** *Let $\mathcal{M}$ be a continual mechanism, and let $\mathcal{Y}$ denote the output space of $\mathcal{M}$ at each step. For $\epsilon \geq 0$ and $0 \leq \delta \leq 1$, $\mathcal{M}$ is said to be (ϵ, δ)-differentially private (or (ϵ, δ)-DP for short) until time step $T \in \mathbb{N}$ if, for every pair of neighboring sequences $\sigma, \sigma' \in \mathcal{X}^T$ and every set of output sequences $V \subseteq \mathcal{Y}^T$, we have*

$$Pr[\mathcal{M}(\sigma) \in V] \leq e^{\epsilon} Pr[\mathcal{M}(\sigma') \in V] + \delta.$$

$\mathcal{M}$ is ϵ-differentially private (or ϵ-DP for short) if $\delta = 0$.

In the above definition, the continual mechanism is differentially private against an *oblivious* adversary. In the *adaptive* setting, however, an *adversary* generates neighboring sequences over time, adaptively, based on the previous outputs of the mechanism. Specifically, given a continual mechanism $\mathcal{M}$, an adversary $\mathcal{A}$ interacts with an *oracle* holding a secret bit $b \in \{0, 1\}$. An oracle is a very limited algorithm that behaves as follows: At each time step, $\mathcal{A}$ issues a pair of updates (u_0, u_1), and the oracle forwards u_b to $\mathcal{M}$. Observing the corresponding output of $\mathcal{M}$, the adversary then adaptively generates the next pair of updates. If, at any time step, the sequences of u_0s and u_1s does not satisfy the required neighboring relation, the oracle halts the interaction.

[2] More generally, $\mathcal{M}$ can be an interleaved sequence of updates and (parametrized) queries, but for simplicity we just assume that one unparametrized query is fixed, and $\mathcal{M}$ answers that query after receiving each update.

We say that the mechanism $\mathcal{M}$ is (ϵ, δ)-differentially private against adaptive adversaries if, for every adversary $\mathcal{A}$, the distributions of output sequences induced by oracle bits $b = 0$ and $b = 1$ are (ϵ, δ)-indistinguishable.

3 Example: Continual Binary Counting

The first and most foundational problem studied in the continual observation model is the *continual binary counting problem*. In this problem, the continual mechanism $\mathcal{M}$ receives a bit x_t at each step $t \in \mathbb{N}$ and returns an estimate y_t of the cumulative sum $s_t = \sum_{i=1}^{t} x_t$. If x_t is an arbitrary real number, the problem is called *continual counting problem*. Two input sequences of the same length are considered to be neighboring if they are identical except for a single time step. The goal is to guarantee differential privacy while providing accuracy, where accuracy is measured as follows:

▶ **Definition 11** ((α, β)-Accuracy for Continual Counter). *For $T \in \mathbb{N}$, $0 < \alpha$ and $0 < \beta \leq 1$, a continual counting mechanism $\mathcal{M}$ is said to be* (α, β)-accurate for T steps *if, for every input sequence $\sigma = (x_1, \ldots, x_T) \in \mathcal{X}^T$,*

$$\Pr[\max_{t \in [T]} |y_t - s_t| \leq \alpha] \geq 1 - \beta,$$

where $s_t = \sum_{i=1}^{t} x_t$ and y_t is the output of $\mathcal{M}$ at step t. We say that $\mathcal{M}$ has additive error α *over the first T steps* with high probability *if it is (α, β)-accurate for T steps, where $\beta = 1/T^c$ for some constant $c > 0$.*

First simple mechanism (Laplace mechanism on each output): A trivial design of a continual counting mechanism $\mathcal{M}$ is to run $T \in \mathbb{N}$ instances of the static Laplace mechanism to privatize the sums s_t for $t \in [T]$. Specifically, at each step $t \in [T]$, the mechanism $\mathcal{M}$ computes s_t and outputs $y_t = s_t + \mathrm{Lap}(1/\epsilon')$, where $\mathrm{Lap}(1/\epsilon')$ is a fresh sample of Laplace noise with scale $1/\epsilon'$. Since changing a single input bit affects each s_t by at most 1, by Lemma 9, the output at each individual step is ϵ'-DP. Thus, applying basic composition (Lemma 4), the sequence of the first T outputs is $T \cdot \epsilon'$-DP. To guarantee $\mathcal{M}$ is ϵ-DP, we must therefore set $\epsilon' = \epsilon/T$. This implies that for each step $t \in [T]$, with a constant probability, $|y_t - s_t| = |\mathrm{Lap}(T/\epsilon)|$ is $\Omega(T/\epsilon)$.

Second simple mechanism (Laplace mechanism on each input) Another trivial approach is to run T instances of Laplace mechanism on the input bits x_t instead of s_t. That is, at each step $t \in [T]$, the continual counting mechanism $\mathcal{M}$ sets $\hat{x}_t = x_t + \mathrm{Lap}(1/\epsilon)$. Since two neighboring input sequences (of length $T \in \mathbb{N}$) differ on at most one coordinate, all but one Laplace mechanism receive the same input for both sequences. Thus, by Lemma 9 and Lemma 4, the sequence $(\hat{x}_1, \ldots, \hat{x}_T)$ is ϵ-DP. Since the output sequence of $\mathcal{M}$ is computed as a post-processing of $(\hat{x}_1, \ldots, \hat{x}_T)$, by Lemma 3, it is distinguishable for two neighboring input sequences. Thus, $\mathcal{M}$ is ϵ-DP. At step T, $|y_T - s_T|$ equals the sum of T i.i.d. samples from $\mathrm{Lap}(1/\epsilon)$. By a standard Chernoff bound, this implies that with high probability, $|y_T - s_T|$ is $\Omega(\sqrt{T})$.

The two naive designs for continual counting illustrate a fundamental trade-off between privacy and accuracy. In the first approach, independent Laplace noise is added to each cumulative sum $s_t = \sum_{i=1}^{t} x_i$. Since each input bit x_i contributes to up to T such sums, satisfying ϵ-differential privacy requires the Laplace scale to be T/ϵ. This, however, results in an additive error that grows linearly with T. In contrast, the second approach adds Laplace noise to each input bit x_t individually and then sums the noisy inputs in a post-processing

step. Here, each input only affects the output of a single Laplace mechanism, allowing for a smaller Laplace scale of $1/\epsilon$. However, because the output at step $t \in [T]$ includes t independent noise, the additive error grows polynomially in t.

To overcome this, Dwork, Naor, Pitassi and Rothblum [19] and Chan, Li, Shi, and Xu [10] proposed a new approach, the so-called *binary (tree) mechanism*, where each input bit is used in the input of $O(\log T)$ Laplace mechanisms, and each output is computed by summing $O(\log T)$ noisy values: Instead of adding noise to each individual bit or to every prefix sum, they add noise to sums of subsequences of the form $(x_{(k-1)\cdot 2^i+1}, \dots, x_{k\cdot 2^i})$ for every $0 \le i \le \log(T)$ and $1 \le k \le T/2^i$. More precisely, at time step $k \cdot 2^i$ (i.e., when the complete subsequence is available), they calculate and store the noisy sum $\sum_{j=(k-1)\cdot 2^i+1}^{k\cdot 2^i} x_j + \text{Lap}(\frac{\log(T)+1}{\epsilon})$. Then, to output the noisy cumulative sum at step $t \in [T]$, they partition the input sequence $(x_1, \dots, x_t)$ into at most $\log(T)$ disjoint subsequences of the above form and return the sum of their noisy values.

Another way to view this dynamic data structure is through a binary tree over time steps 1 to T, where each node stores the noisy sum of the corresponding bits in its subtree, i.e., runs the Laplace mechanism on this partial sum. At each level of the tree, the input bits are partitioned between the nodes. Thus, the (parallel) composition of Laplace mechanisms at each level is $\frac{\epsilon}{\log(T)+1}$-DP. Composing the $\log(T)+1$ levels and applying the basic composition theorem, we conclude that the joint output of all Laplace mechanisms is ϵ-DP. Since the noisy cumulative sum at every step is a post-processing of these noisy partial sums, this implies that the continual counter is ϵ-DP.

For the accuracy analysis, note that each output equals the true cumulative sum plus at most $\log(T)$ i.i.d. noise drawn from $\text{Lap}\left(\frac{\log(T)+1}{\epsilon}\right)$. Chan et al. [10] showed that the sum of these $\log(T)$ Laplace random variables is $O(\sqrt{\log(T)}\frac{\log(T)+1}{\epsilon}\sqrt{\log(1/\beta)})$ with probability at least $1-\beta$. Applying a union bound over all T time steps, the maximum additive error across all steps is $O(\log(T)^{3/2}\sqrt{\log(T/\beta)}/\epsilon)$ with probability at least $1-\beta$.

4 Interactive Setting

Consider a hospital that collected patient records during the Covid-19 pandemic, i.e., it has a static dataset. Over time, researchers may issue a sequence of statistical queries to investigate long-term effects of Covid and its potential correlation with current medical conditions. These queries are *parametrized*, meaning that they are given one or multiple parameters determining which query is asked, and these parameters are often chosen *adaptively*, meaning that the parameters for each new query may be chosen based on the responses to previous ones. To protect each patient record, queries must satisfy certain properties, e.g., an analyst cannot ask whether Mr. X had Covid or not.

In the DP literature, a mechanism that holds a static dataset and answers multiple adaptive queries is usually referred to as an *interactive mechanism*. To formally define differential privacy for the interactive setting, we need to define a *view* random variable: Let $\mathcal{M}$ be an interactive mechanism, and let $\mathcal{A}$ be an analyst (or adversary) that adaptively asks queries from $\mathcal{M}$. We denote the mechanism $\mathcal{M}$ holding dataset D by $\mathcal{M}(D)$. The *view* of the analyst $\mathcal{A}$ interacting with $\mathcal{M}(D)$, denoted $\text{View}(\mathcal{A}, \mathcal{M}(D))$, is the sequence $(r_{\mathcal{A}}, q_1, a_1, q_2, a_2, \dots)$ consisting of the analyst's internal randomness $r_{\mathcal{A}}$ and the sequence of queries q_i and corresponding answers a_i returned by $\mathcal{M}$.

▶ **Definition 12.** *An interactive mechanism $\mathcal{M}$ is said to be (ϵ, δ)-differentially private (or (ϵ, δ)-DP for short) if for every adversary $\mathcal{A}$ and every pair of neighboring datasets D_0 and D_1, the random variables View$(\mathcal{A}, \mathcal{M}(D_0))$ and View$(\mathcal{A}, \mathcal{M}(D_1))$ are (ϵ, δ)-indistinguishable. We say that $\mathcal{M}$ is ϵ-differentially private (or ϵ-DP for short) if $\delta = 0$.*

A fundamental problem in the interactive setting is *threshold testing*, where the mechanism holds a static dataset $D \in \mathrm{MS}(\{0,1\})$ consisting of binary values, and receives a sequence of integer thresholds over time. Specifically, at each time step $t \in \mathbb{N}$, the analyst sends a threshold query $\texttt{thresh}_t \in \mathbb{N}$, asking whether the sum $\sum_{x \in D} x$ exceeds the given threshold $\texttt{thresh}_t$. The mechanism responds with either $\top$ (yes) or $\bot$ (no). Once the mechanism returns $\top$, it is required to halt. For simplicity, we assume the mechanism continues to return $\top$ indefinitely after it outputs $\top$ for the first time.

Two datasets $D_0, D_1 \in \mathrm{MS}(\{0,1\})$ are considered *neighboring* if they differ in the presence or absence of one element. The goal is to provide accurate answers to threshold queries while preserving privacy, where accuracy is defined as follows:

▶ **Definition 13.** *A threshold testing mechanism $\mathcal{M}$ is said to be (α, β)-accurate until step $T \in \mathbb{N}$ if, for every input dataset $D \in MS(\{0,1\})$, with probability at least $1-\beta$, the following conditions hold for every $t \in [T]$:*

1. ***(Above threshold).*** *If $\sum_{x \in D} x \geq \texttt{thresh}_t + \alpha$, then $\mathcal{M}(D)$ returns $\top$ at step t.*
2. ***(Below threshold).*** *If for every $i \leq t$, $\sum_{x \in D} x \leq \texttt{thresh}_i - \alpha$, then $\mathcal{M}(D)$ returns $\bot$ at all steps $i \leq t$.*

A well-known continual threshold-testing mechanism is the sparse vector technique (SVT) mechanism [19, 36]. The SVT mechanism allows an analyst to issue adaptive threshold queries on a static dataset while preserving differential privacy. This mechanism is described in Algorithms 1 and 2.

In Algorithm 1, the mechanism is initialized with a dataset D and a privacy parameter ϵ. It first computes the true count $s = \sum_{x \in D} x$ and then draws a sample Z_0 from $\mathrm{Lap}(2/\epsilon)$. This sample will be repeatedly used in all future comparisons. Upon receiving a threshold query $\texttt{thresh}_t$, the mechanism proceeds as follows (see Algorithm 2): If it has already returned $\top$ in any prior step, it returns $\top$. Otherwise, using the old noise Z_0 and a fresh noise $Z_t \sim \mathrm{Lap}(4/\epsilon)$, it compares $s + Z_0 \geq \texttt{thresh}_t + Z_t$. If the inequality holds, then the mechanism returns $\top$ and sets a flag to indicate that it has exceeded a threshold; otherwise, it returns $\bot$.

▶ **Lemma 14** (Privacy [19] and Accuracy [35] of SVT)**.** *For every $\epsilon \geq 0$, the SVT mechanism described in Algorithms 1 and 2 satisfies ϵ-DP. Moreover, for every $0 < \beta \leq 1$ and $T \in \mathbb{N}$, this mechanism is $(\alpha = O(\log(T/\beta)/\epsilon), \beta)$-accurate until step T.*

Algorithm 1 $\texttt{svt}_\epsilon$ (initialization).

Input: privacy parameter $\epsilon > 0$; dataset $D \in \mathrm{MS}(\{0,1\})$.

1: $\epsilon_1 = \epsilon_2 = \epsilon/2$
2: $s = \sum_{x \in D} x$
3: $\texttt{varflag} = 0$
4: Draw $Z_0 \sim \mathrm{Lap}(1/\epsilon_1)$

Algorithm 2 $\texttt{svt}_\epsilon$ (update).

Input: threshold $\texttt{thresh}_t$
Output: an element of $\{\top, \bot\}$.
if $\texttt{varflag} = 1$ **then**
  return $\top$
end if
Draw $Z_t \sim \text{Lap}(2/\epsilon_2)$
if $s + Z_0 \geq \texttt{thresh}_t + Z_t$ **then**
  $\texttt{varflag} = 1$
  return $\top$
else
  return $\bot$
end if

5 Designing New Continual Mechanisms

In this section, we introduce new composition tools that enable a modular construction of complex continual mechanisms executing not only non-interactive mechanisms (NIMs) but also continual and interactive mechanisms as sub-mechanisms. As we mentioned before, the composition theorems presented in Section 1 only apply to NIMs. This raises a question: can interactive or continual sub-mechanisms always be decomposed into NIMs followed by post-processing, thereby enabling privacy analysis of the complex mechanism using composition theorems for NIMs and the post-processing lemma? We will first show that such a decomposition is not always possible and then present composition tools that simplify the privacy analysis of the mechanism with modular design. Due to complicated dependencies between the outputs and inputs of the sub-mechanisms and their potential influence on each other, analyzing the privacy of these mechanisms from scratch can be challenging.

Roughly speaking, interactive mechanisms can be viewed as a special case of continual mechanisms, where the first input update is a dataset and all subsequent updates are adaptively chosen queries with no effect on the dataset. When modeling an interactive mechanism as a continual mechanism, two input sequences are considered neighboring if their first element (i.e., the dataset) are neighboring, and all subsequent elements (i.e., the queries) are identical. This perspective allows us to unify continual and interactive mechanisms and refer to both of them as *continual mechanisms (CMs)*.

Continual mechanisms are not necessarily compositions of non-interactive mechanisms. We begin by demonstrating that an (ϵ, δ)-DP continual mechanism cannot, in general, be expressed as the post-processing of a composition of NIMs, each using independent random coins, such that the composition satisfies (ϵ, δ)-DP. To illustrate this, we compare the privacy analysis of the binary-tree mechanism (Section 3) and of the Sparse Vector Technique (SVT) mechanism (Section 4).

Let $T \in \mathbb{N}$ and $\epsilon \geq 0$. The binary-tree mechanism draws $2T - 1$ independent Laplace noises from $\text{Lap}((\log T + 1)/\epsilon)$, one for each node of the binary tree. Each noise is then used individually to privatize a partial sum of the input sequence, and the output at each time step is computed by combining a subset of these privatized values. This structure enables a clean decomposition of the mechanism into NIMs with independent randomness, followed by a post-processing.

In contrast, the privacy analysis of the SVT mechanism is different: given a dataset D, this mechanism samples $T+1$ independent Laplace noises: $Z_0 \sim \text{Lap}(2/\epsilon)$ and $Z_i \sim \text{Lap}(4/\epsilon)$ for each $i \in [T]$. At time step $t \in [T]$, given a threshold $\texttt{thresh}_t$, the mechanism determines the output based on the comparison $\sum_{x \in D} x + Z_t \geq \texttt{thresh}_t + Z_0$. Each noisy sum $\sum_{x \in D} x + Z_t$ can be considered as an instance of the Laplace mechanism with the privacy parameter $\epsilon/4$, and the comparison can be interpreted as a post-processing of this $\epsilon/4$-DP mechanism. However, since each data record in D contributes to the input of all T Laplace mechanisms, the composition of these mechanisms would be $\frac{T}{4\epsilon}$-DP.

To avoid this undesirable polynomial privacy parameter, the SVT mechanism is analyzed directly by comparing the view of any adversary interacting with this mechanism for any two neighboring datasets. This analysis yields an ϵ-DP guarantee. Notably, this proof crucially exploits a shared source of randomness, i.e., the random variable Z_0, to get a better privacy guarantee, something that the privacy analysis by composition cannot capture as it requires the composed mechanisms to use independent random coins.

Composition theorems for continual mechanisms. This limitation implies that, in general, when analyzing a mechanisms executing multiple continual sub-mechanisms, we cannot simply decompose each continual sub-mechanism into NIMs and then apply standard composition theorems for NIMs to derive overall privacy guarantees. This challenge has led to a line of research on extending the composition theorems for NIMs to *concurrent composition* theorems for CMs [43, 37, 44, 27, 28]. Here, the term *concurrent* captures the fact that the inputs to the composed CMs are generated concurrently: Specifically, the input to one mechanism may depend on the outputs of *all* other mechanisms up to that point.

Although we do not formally define concurrent composition in this paper, we highlight a recent work by Henzinger, Safavi, and Vadhan [28] showing that composition theorems for NIMs, such as Theorem 4 and Theorem 5, still hold in the more general setting of the concurrent composition of CMs, provided that the composed mechanisms are differentially private against *adaptive* adversaries.

Moreover, they show a separation between parallel composition in the non-interactive model (Theorem 6) and the concurrent parallel composition of CMs, where an adversary adaptively selects which mechanism receives non-identical input sequences. To address this, they develop new concurrent parallel composition theorems.

Assume a mechanism $\mathcal{M}$ uses multiple CMs as subroutines. Henzinger et al. [28] also formalize which types of *post-processing* on the outputs of these sub-mechanisms are permissible in order to preserve the privacy guarantee of the concurrent composition of the sub-mechanisms: They present a general framework for combining the given inputs and the previous outputs of sub-mechanisms to generate new inputs for the sub-mechanisms and then generate the final output based on the outcomes of the sub-mechanisms. This formalism results in a simple privacy analysis for complex mechanisms executing continual sub-mechanisms.

6 Conclusions

Continual mechanisms are the analogue of dynamic data structures in the differentially private setting. Apart from analyzing their accuracy, time, and space requirements, their privacy properties need to be proven, which can be challenging. New composition theorems for continual mechanisms simplify this task and will hopefully support the design of continual mechanisms for various applications with improved tradeoffs between privacy and accuracy.

References

1 Joel Daniel Andersson and Rasmus Pagh. Streaming private continual counting via binning. *CoRR*, abs/2412.07093, 2024. doi:10.48550/arXiv.2412.07093.

2 Joel Daniel Andersson, Rasmus Pagh, and Sahel Torkamani. Improved counting under continual observation with pure differential privacy. *CoRR*, abs/2408.07021, 2024. doi:10.48550/arXiv.2408.07021.

3 Apple. Privacy-preserving contact tracing, https://covid19.apple.com/contacttracing, 2021.

4 Jean Bolot, Nadia Fawaz, S. Muthukrishnan, Aleksandar Nikolov, and Nina Taft. Private decayed sum estimation under continual observation. *CoRR*, abs/1108.6123, 2011. arXiv:1108.6123.

5 Mark Bun, Marco Gaboardi, Marcel Neunhoeffer, and Wanrong Zhang. Continual release of differentially private synthetic data from longitudinal data collections. *Proc. ACM Manag. Data*, 2(2):94, 2024. doi:10.1145/3651595.

6 Mark Bun, Kobbi Nissim, Uri Stemmer, and Salil Vadhan. Differentially private release and learning of threshold functions. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 634–649. IEEE, 2015. doi:10.1109/FOCS.2015.45.

7 Adrian Rivera Cardoso and Ryan Rogers. Differentially private histograms under continual observation: Streaming selection into the unknown. In *Proc. 25th AISTATS*, pages 2397–2419, 2022. URL: https://proceedings.mlr.press/v151/rivera-cardoso22a.html.

8 Ho-Leung Chan, Tak-Wah Lam, Lap-Kei Lee, and Hing-Fung Ting. Continuous monitoring of distributed data streams over a time-based sliding window. *Algorithmica*, 62(3-4):1088–1111, 2012. doi:10.1007/S00453-011-9506-5.

9 T-H Hubert Chan, Mingfei Li, Elaine Shi, and Wenchang Xu. Differentially private continual monitoring of heavy hitters from distributed streams. In *International Symposium on Privacy Enhancing Technologies Symposium*, pages 140–159. Springer, 2012. doi:10.1007/978-3-642-31680-7_8.

10 T.-H. Hubert Chan, Elaine Shi, and Dawn Song. Private and continual release of statistics. *ACM Trans. Inf. Syst. Secur.*, 14(3):26:1–26:24, 2011. doi:10.1145/2043621.2043626.

11 Rachel Cummings, Sara Krehbiel, Kevin A. Lai, and Uthaipon Tao Tantipongpipat. Differential privacy for growing databases. In *Proc. 31st NeurIPS*, pages 8878–8887, 2018. URL: https://proceedings.neurips.cc/paper/2018/hash/ac27b77292582bc293a51055bfc994ee-Abstract.html.

12 Aref N. Dajani, Amy D. Lauger, Phyllis E. Singer1, Daniel Kifer, Jerome P. Reiter, Ashwin Machanavajjhala, Simson L. Garfinkel, Scot A. Dahl, Matthew Graham Vishesh Karwa, Hang Kim, Philip Leclerc, Ian M. Schmutte, William N. Sexton, Lars Vilhuber, and John M. Abowd. The modernization of statistical disclosure limitation at the U.S. Census Bureau, 2017. URL: https://www2.census.gov/cac/sac/meetings/2017-09/statistical-disclosure-limitation.pdf.

13 Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. Collecting telemetry data privately. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 3571–3580, 2017. URL: http://papers.nips.cc/paper/6948-collecting-telemetry-data-privately.

14 Wei Dong, Zijun Chen, Qiyao Luo, Elaine Shi, and Ke Yi. Continual observation of joins under differential privacy. *Proceedings of the ACM on Management of Data*, 2(3):1–27, 2024.

15 Max Dupré la Tour, Monika Henzinger, and David Saulpic. Making old things new: a unified algorithm for differentially private clustering. In *Proc. 41th ICML*, 2024.

16 Krishnamurthy Dj Dvijotham, H. Brendan McMahan, Krishna Pillutla, Thomas Steinke, and Abhradeep Thakurta. Efficient and near-optimal noise generation for streaming differential privacy. In *65th IEEE Annual Symposium on Foundations of Computer Science, FOCS*

2024, Chicago, IL, USA, October 27-30, 2024, pages 2306–2317. IEEE, 2024. doi:10.1109/FOCS61266.2024.00135.

17 Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3*, pages 265–284. Springer, 2006. doi:10.1007/11681878_14.

18 Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N. Rothblum. Differential privacy under continual observation. In *Proc. of the Forty-Second ACM Symp. on Theory of Computing (STOC'10)*, pages 715–724, 2010. doi:10.1145/1806689.1806787.

19 Cynthia Dwork, Moni Naor, Omer Reingold, Guy N Rothblum, and Salil Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 381–390, 2009. doi:10.1145/1536414.1536467.

20 Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014. doi:10.1561/0400000042.

21 Cynthia Dwork, Guy N. Rothblum, and Salil Vadhan. Boosting and differential privacy. In *Proceedings of the IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 51–60, 2010. doi:10.1109/FOCS.2010.12.

22 Alessandro Epasto, Quanquan C. Liu, Tamalika Mukherjee, and Felix Zhou. The power of graph sparsification in the continual release model. *CoRR*, abs/2407.17619, 2024. doi:10.48550/arXiv.2407.17619.

23 Alessandro Epasto, Jieming Mao, Andres Muñoz Medina, Vahab Mirrokni, Sergei Vassilvitskii, and Peilin Zhong. Differentially private continual releases of streaming frequency moment estimations. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference, ITCS 2023, January 10-13, 2023, MIT, Cambridge, Massachusetts, USA*, volume 251 of *LIPIcs*, pages 48:1–48:24. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICS.ITCS.2023.48.

24 Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, pages 1054–1067. ACM, 2014. doi:10.1145/2660267.2660348.

25 Hendrik Fichtenberger, Monika Henzinger, and Wolfgang Ost. Differentially private algorithms for graphs under continual observation. In *29th Annual European Symposium on Algorithms, ESA*, 2021. doi:10.4230/LIPIcs.ESA.2021.42.

26 Hendrik Fichtenberger, Monika Henzinger, and Jalaj Upadhyay. Constant matters: Fine-grained error bound on differentially private continual observation. In *International Conference on Machine Learning*, pages 10072–10092. PMLR, 2023. URL: https://proceedings.mlr.press/v202/fichtenberger23a.html.

27 Samuel Haney, Michael Shoemate, Grace Tian, Salil Vadhan, Andrew Vyrros, Vicki Xu, and Wanrong Zhang. Concurrent composition for interactive differential privacy with adaptive privacy-loss parameters. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pages 1949–1963, 2023. doi:10.1145/3576915.3623128.

28 Monika Henzinger, Roodabeh Safavi, and Salil Vadhan. Concurrent composition for differentially private continual mechanisms. *arXiv preprint*, 2024. arXiv:2411.03299.

29 Monika Henzinger, A. R. Sricharan, and Teresa Anna Steiner. Private counting of distinct elements in the turnstile model and extensions. In Amit Kumar and Noga Ron-Zewi, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2024, August 28-30, 2024, London School of Economics, London, UK*, volume 317 of *LIPIcs*, pages 40:1–40:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.APPROX/RANDOM.2024.40.

30 Monika Henzinger, A. R. Sricharan, and Teresa Anna Steiner. Differentially private continual release of histograms and related queries. In *The 28th International Conference on Artificial Intelligence and Statistics*, 2025. URL: https://openreview.net/forum?id=EYhGcfYpFS.

31 Monika Henzinger and Jalaj Upadhyay. Improved differentially private continual observation using group algebra. In Yossi Azar and Debmalya Panigrahi, editors, *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2025, New Orleans, LA, USA, January 12-15, 2025*, pages 2951–2970. SIAM, 2025. doi:10.1137/1.9781611978322.95.

32 Monika Henzinger, Jalaj Upadhyay, and Sarvagya Upadhyay. Almost tight error bounds on differentially private continual counting. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 5003–5039, 2023. doi:10.1137/1.9781611977554.CH183.

33 Monika Henzinger, Jalaj Upadhyay, and Sarvagya Upadhyay. A unifying framework for differentially private sums under continual observation. In David P. Woodruff, editor, *Proceedings of the 2024 ACM-SIAM Symposium on Discrete Algorithms, SODA 2024, Alexandria, VA, USA, January 7-10, 2024*, pages 995–1018. SIAM, 2024. doi:10.1137/1.9781611977912.38.

34 Palak Jain, Iden Kalemaj, Sofya Raskhodnikova, Satchit Sivakumar, and Adam D. Smith. Counting distinct elements in the turnstile model with differential privacy under continual observation. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL: http://papers.nips.cc/paper_files/paper/2023/hash/0ef1afa0daa888d695dcd5e9513bafa3-Abstract-Conference.html.

35 Palak Jain, Adam D. Smith, and Connor Wagaman. Time-aware projections: Truly node-private graph statistics under continual observation. In *IEEE Symposium on Security and Privacy, SP 2024, San Francisco, CA, USA, May 19-23, 2024*, pages 127–145. IEEE, 2024. doi:10.1109/SP54263.2024.00196.

36 Min Lyu, Dong Su, and Ninghui Li. Understanding the sparse vector technique for differential privacy. *arXiv preprint arXiv:1603.01699*, 2016. arXiv:1603.01699.

37 Xin Lyu. Composition theorems for interactive differential privacy. *Advances in Neural Information Processing Systems*, 35:9700–9712, 2022.

38 Frank D McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, pages 19–30, 2009. doi:10.1145/1559845.1559850.

39 Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In *Security and Privacy, 2008. SP 2008. IEEE Symposium on*, pages 111–125. IEEE, 2008. doi:10.1109/SP.2008.33.

40 Yuan Qiu and Ke Yi. Differential privacy on dynamic data. *arXiv preprint arXiv:2209.01387*, 2022. doi:10.48550/arXiv.2209.01387.

41 Sofya Raskhodnikova and Teresa Anna Steiner. Fully dynamic algorithms for graph databases with edge differential privacy. *Proceedings of the ACM on Management of Data*, 3(2):1–28, 2025. doi:10.1145/3725236.

42 Shuang Song, Susan Little, Sanjay Mehta, Staal Vinterbo, and Kamalika Chaudhuri. Differentially private continual release of graph statistics. *arXiv preprint*, 2018. arXiv:1809.02575.

43 Salil Vadhan and Tianhao Wang. Concurrent composition of differential privacy. In *Theory of Cryptography: 19th International Conference, TCC 2021, Raleigh, NC, USA, November 8–11, 2021, Proceedings, Part II 19*, pages 582–604. Springer, 2021. doi:10.1007/978-3-030-90453-1_20.

44 Salil Vadhan and Wanrong Zhang. Concurrent composition theorems for differential privacy. In *55th Annual ACM Symposium on Theory of Computing*, 2023.

45 Tianhao Wang, Joann Chen, Zhikun Zhang, Dong Su, Yueqiang Cheng, Zhou Li, Ninghui Li, and Somesh Jha. Continuous release of data streams under both centralized and local differential privacy. In *CCS*, pages 1237–1253, 2021.

46 Dongdong Xie, Pinghui Wang, Quanqing Xu, Chuanhui Yang, and Rundong Li. Efficient and accurate differentially private cardinality continual releases. *Proceedings of the ACM on Management of Data*, 3(3):1–27, 2025. doi:10.1145/3725288.

47 Timothy Yang, Galen Andrew, Hubert Eichner, Haicheng Sun, Wei Li, Nicholas Kong, Daniel Ramage, and Françoise Beaufays. Applied federated learning: Improving google keyboard query suggestions. *arXiv preprint arXiv:1812.02903*, 2018. arXiv:1812.02903.

48 Bing Zhang, Vadym Doroshenko, Peter Kairouz, Thomas Steinke, Abhradeep Thakurta, Ziyin Ma, Eidan Cohen, Himani Apte, and Jodi Spacek. Differentially private stream processing at scale. *Proc. VLDB Endow.*, 17(12):4145–4158, 2024. doi:10.14778/3685800.3685833.