Private and Continual Release of Statistics

T-H. HUBERT CHAN, The University of Hong Kong
ELAINE SHI, Palo Alto Research Center
DAWN SONG, UC Berkeley

We ask the question – *how can websites and data aggregators continually release updated statistics, and meanwhile preserve each individual user's privacy?* Suppose we are given a stream of 0's and 1's. We propose a differentially private continual counter that outputs at every time step the approximate number of 1's seen thus far. Our counter construction has error that is only poly-log in the number of time steps. We can extend the basic counter construction to allow websites to continually give top-k and hot items suggestions while preserving users' privacy.

Categories and Subject Descriptors: F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Non-numerical Algorithms and Problems—*Computations on discrete structures*

General Terms: Privacy, Algorithms

Additional Key Words and Phrases: Differential privacy, continual mechanism, streaming algorithm

1. INTRODUCTION

Websites such as online retailers, search engines and social networks commonly publish aggregate statistics about their users to realize valuable social and economic utilities. Moreover, the published statistics are continually updated over time as new data arrive. Such practices are ubiquitous and we name a few examples below. Sites such as Amazon, IMDB, Delicious and Flickr recommend popular items or content to users to enhance their browsing experience and engage their interests. Search engines such as Google and Yahoo help a user to auto-complete her search query by suggesting the most frequent search terms matching the prefix specified by the user. During political campaigns, websites survey the population and continually update the support rates for candidates.

Releasing aggregate information about users may seem harmless at first glance. However, previous work has shown that such statistical disclosures can expose sensitive information about an individual user [Dinur and Nissim 2003; Dwork and Yekhanin 2008]. In particular, sites that continually update the published statistics over time can give even more leverage to the adversary and result in more severe privacy leakage [Calandrino et al. 2009].

In this paper, we ask the question – *how can we guarantee the users' privacy when a website must continually publish new statistics as new data arrive?* Independent from our work, Dwork *et.al.* also consider essentially the same problem, and they phrase the problem as *"differential privacy under continual observation"* [Dwork 2010a; Dwork et al. 2010; Dwork et al. 2010].

This work is partially supported by the National Science Foundation under Grants No. 0716230, 0448452 and CCF-0424422, and by the Office of Naval Research under MURI Grant No. N000140911081. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation, or the Office of Naval Research.

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

The setting we consider is different from the traditional setting in which differential privacy was studied. The traditional setting assumes a static input database, and a curator who must answer k interactive queries or publish some sanitized statistics of the database non-interactively. In our setting, the input database is dynamic and evolves over time, and a mechanism must update the published statistics as new data items arrive. Therefore, traditional differentially private mechanisms either fail to apply directly to our setting, or result in an unsatisfactory loss in terms of utility or privacy if applied naively.

1.1. Contributions

Differentially private continual counter with poly-log error. We consider the *continual counting problem*. Assume that the input stream $\sigma \in \{0,1\}^{\mathbb{N}}$ is a sequence of bits. The bit $\sigma(t)$ at time $t \in \mathbb{N}$ may denote whether an event of interest occurred at time t, e.g., whether a user purchased an item at time t. At every time step $t \in \mathbb{N}$, the mechanism must output an approximate count of the number of 1's seen thus far.

We design an ϵ-differentially private continual counter with small error. Specifically, for each $t \in \mathbb{N}$, with probability at least $1-\delta$, we guarantee $O(\frac{1}{\epsilon} \cdot (\log t)^{1.5} \cdot \log \frac{1}{\delta})$ error[1]. In an independent work by Dwork *et.al.* [Dwork et al. 2010], they also show a similar upper bound of $O(\frac{1}{\epsilon} \cdot (\log t)^{1.5})$ (omitting the δ term). The above upper bound is almost tight, since Dwork *et.al.* [Dwork et al. 2010] show that any ϵ-differentially private mechanism will for some stream, with probability at least δ, make an error of at least $\Omega(\frac{1}{\epsilon}(\log T + \log \frac{1}{\delta}))$ at some time before T.

Our mechanism achieves *time unboundedness*, i.e., the mechanism does not require a priori knowledge of an upper bound on the time for which it will run, and provides guarantees even when it is run indefinitely. This represents an improvement over the work by Dwork *et.al.* [Dwork 2010a] – to the best of our knowledge, their mechanism needs to know an upper bound on the number of time steps.

Pan privacy. Dwork *et.al.* first introduced the notion of pan privacy [Dwork et al. 2010; Dwork 2010a]. A mechanism is pan private if it can preserve differential privacy even when an adversary can observe snapshots of the mechanism's internal states, e.g., in subpoenas. We show how to modify our mechanism to achieve pan privacy, without incurring any loss in the asymptotic guarantees (Section 6).

Consistency. Some applications require that the mechanism is "consistent", i.e., there must exist a valid stream that agrees with the output of the mechanism. This means that the mechanism must output integer counts, and furthermore, the count must increase by either 0 or 1 with each time step. In Section 5, we show how to transform our mechanism to achieve consistency such that with high probability, for all time steps t the resulting error bound is $O(\frac{1}{\epsilon} \cdot (\log t)^{2.5})$.

Applications. Our continual counter construction has immediate practical applications. As mentioned earlier, it is a common practice for websites to suggest to users the most popular movies, news items or photos. In Section 7, we show how websites can continually make such top-k or hot items suggestions in a differentially private manner. Moreover, we show that our techniques can be generalized to construct a differentially private mechanism that answers multi-dimensional range queries.

The counter is also an important primitive in numerous data streaming algorithms [Metwally et al. 2005; Demaine et al. 2002; Manku and Motwani 2002]. Our differentially private continual counter is an initial step towards designing a broad class of streaming algorithms that continually report outputs over time.

[1]For large values of t, we can actually get a better bound $O(\frac{1}{\epsilon} \cdot (\log t)^{1.5} \cdot \sqrt{\log \frac{1}{\delta}})$. To get a high probability statement, we can set $\delta := \frac{1}{\mathrm{poly}(t)}$ and the corresponding error becomes $O((\log t)^2/\epsilon)$.

1.2. Related Work

Most closely related work. Independent from our work, Dwork *et.al.* show a similar result in a recent paper [Dwork et al. 2010]. They also construct a differentially private continual counter with error $O(\frac{1}{\epsilon} \cdot (\log t)^{1.5})$ where t is the number of timesteps. Moreover, they show a lower bound of $\Omega(O(\frac{1}{\epsilon} \cdot (\log t))$, indicating that the upper bound is almost tight. A preliminary version of the result was revealed at the SODA'10 conference [Dwork 2010a] in an invited talk by Dwork. The preliminary result contains a slightly looser upper bound – a differentially private continual counter with error square root in the number of 1's seen thus far.

Xiao *et.al.* [Xiao et al. 2010] use wavelet transforms to achieve differential privacy for databases. Although their motivation is different from ours, their construction uses a similar principle to our Binary Mechanism (Section 3.4), and hence they can also achieve the same error bounds as this paper and [Dwork et al. 2010]. However, they only consider the offline setting. They extend the wavelet framework to multi-dimensional databases, while we extend our Binary Mechanism for the same application in Section 7 to achieve the same error bounds. In terms of error bounding techniques, they consider the variance of the error involved, while we prove comprehensive high probability statements concerning the error using measure concentration techniques.

The offline setting of our problem is a special case of linear counting queries (each of a query is simply the sum of bits from a prefix of the stream). Li *et.al.* [Li et al. 2010] use linear algebra techniques to answer linear queries in a differentially private manner. When applied to our problem, their techniques also achieve the same error bound.

Dwork, Naor, Pitassi and Rothblum [Dwork et al. 2010] recently propose the notion of pan privacy, i.e., how to achieve differential privacy even in the presence of intrusions, in which the adversary is allowed access to the mechanism's internal states. Dwork *et.al.* used the notion of pan privacy in the continual counter mechanism [Dwork et al. 2010; Dwork 2010a], and showed how to make their counter mechanism resilient against a single unannounced intrusion. Inspired by their techniques, we also convert our mechanism to a pan private version that is immune to a single unannounced intrusion or multiple afterwards announced intrusions.

Differential privacy in the traditional setting. In the traditional setting, a trusted curator who holds a large data set must respond to queries *interactively* or publish sanitized statistics about the data *non-interactively*. The notion of differential privacy was first proposed and studied by Dwork *et.al.* [Dwork 2006; Dwork et al. 2006]. An extensive literature has since emerged, studying the different tradeoffs between utility and privacy. To better understand the motivation and state-of-the-art of this line of research, we recommend the readers to these excellent survey papers by Dwork [Dwork 2009; 2010b; 2008].

Researchers have also applied theoretical results in differential privacy to real-world applications. For example, McSherry and Mironov show how to build privacy into the Netflix database published for the Netflix contest [McSherry and Mironov 2009]. Korolova *et.al.* show how to release search logs and click logs privately [Korolova et al. 2009].

Consistency. Hay *et.al.* [Hay et al. 2010] consider boosting the accuracy of differentially private histogram via consistency. Their idea is to first construct a differentially private output, and release the closest output that satisfies some consistency rules (such as monotonicity). They show that this strategy can reduce the error for some cases where the output is a sorted sequence of numbers. However, in their application, the sensitivity of the output is constant, which is not the case for our problem, because changing the first bit of the stream will change the answer for every time step. Our approach to achieve consistency is different. We first use the Binary Mechanism to achieve low error, and in Section 5 we give a simple rounding procedure to achieve consistency with the same error bound.

Attacks against privacy. A complementary line of research is attacks against privacy. Narayanan *et.al.* show how to de-anonymize the Netflix data set [Narayanan and Shmatikov 2008]. Jones *et.al.* show how to break the privacy of query log bundles [Jones et al. 2008]. More relevant to this work, Calandrino *et.al.* [Calandrino et al. 2009] recently demonstrate that by observing continual updates from websites such as Amazon over a period of time, an adversary can learn individual user behavior at a fine level of granularity. Our work is partly inspired by the problem they expose.

2. PRELIMINARIES

2.1. Continual Counting Mechanism

We consider streams of 0's and 1's. Formally, a stream $\sigma \in \{0,1\}^{\mathbb{N}}$ is a bit-string of countable length, where $\mathbb{N} := \{1,2,3,\ldots\}$ is the set of positive integers. Specifically, $\sigma(t) \in \{0,1\}$ denotes the bit at time $t \in \mathbb{N}$. We write $[T] := \{1,2,3,\ldots,T\}$ and $\sigma_T \in \{0,1\}^T$ is the length T prefix of the stream σ. We will use the term *item* to refer to a bit in the stream.

At every time t, we wish to output the number of 1's that have arrived up to time t.

Definition 2.1 (*Continual Counting Query*). Given a stream $\sigma \in \{0,1\}^{\mathbb{N}}$, the count for the stream is a mapping $c_\sigma : \mathbb{N} \to \mathbb{Z}$ such that for each $t \in \mathbb{N}$, $c_\sigma(t) := \sum_{i=1}^{t} \sigma(i)$. We write c instead of c_σ when there is no risk of ambiguity on the stream σ in question.

We now formally define the notion of a continual counting mechanism which continually outputs the number of 1's seen thus far.

Definition 2.2 (*Counting Mechanism*). A counting mechanism $\mathcal{M}$ takes a stream $\sigma \in \{0,1\}^{\mathbb{N}}$ and produces a (possibly randomized) mapping $\mathcal{M}(\sigma) : \mathbb{N} \to \mathbb{R}$. Moreover, for all $t \in \mathbb{N}$, $\mathcal{M}(\sigma)(t)$ is independent of all $\sigma(i)$'s for $i > t$. We can also view $\mathcal{M}(\sigma)$ as a point in $\mathbb{R}^{\mathbb{N}}$. When there is no risk of ambiguity on the stream σ in question, we drop the dependence on σ and use $\mathcal{M}(t)$ to mean $\mathcal{M}(\sigma)(t)$.

Definition 2.3 (*Time-bounded Mechanism*). A counting mechanism $\mathcal{M}$ is unbounded, if it accepts streams of indefinite lengths, i.e., given any stream σ, $\mathcal{M}(\sigma) \in \mathbb{R}^{\mathbb{N}}$. Given $T \in \mathbb{N}$, a mechanism $\mathcal{M}$ is T-bounded if it only accepts streams of lengths at most T and returns $\mathcal{M}(\sigma) \in \mathbb{R}^T$. In other words, the mechanism needs to know the value T in advance and only looks at the length T prefix of any given stream.

We would like the mechanism to be useful, that is, its output should well approximate the true count at any point of time. We formally define the notion of utility below.

Definition 2.4 (*Utility*). A counting mechanism $\mathcal{M}$ is (λ, δ)-useful at time t, if for any stream σ, with probability (over the randomness of $\mathcal{M}$) at least $1 - \delta$, we have $|c_\sigma(t) - \mathcal{M}(\sigma)(t)| \leq \lambda$. Note that λ may be a function of δ and t.

Remark 2.5. Our definition of utility covers the usefulness of the mechanism for a single timestep. A standard union bound argument can be applied if the utility for multiple timesteps are required.

2.2. Differential Privacy

Intuitively, a mechanism is differentially private if it cannot be used to distinguish two streams that are almost the same. In other words, an adversary is unable to determine whether an event of interest took place or not by observing the output of the mechanism over time. For example, the adversary is unable to determine whether a user purchased an item at some time t.

Definition 2.6 (*Differential Privacy*). Two streams σ and σ' are adjacent if they differ at exactly one time t. A (randomized) counting mechanism $\mathcal{M}$ is ϵ-differentially private (or preserves ϵ-differential privacy) if for any adjacent streams σ and σ', and any measurable subset $S \subseteq \mathbb{R}^{\mathbb{N}}$ (or $S \subseteq \mathbb{R}^T$ for T-bounded mechanisms), $\Pr[\mathcal{M}(\sigma) \in S] \leq \exp(\epsilon) \cdot \Pr[\mathcal{M}(\sigma') \in S]$.

Remark 2.7. In [Dwork et al. 2010], a stream of objects from some set X are considered. In their setting, a stream σ is a point in $X^{\mathbb{N}}$. For user-level privacy, two streams are considered to be adjacent if they differ only in the presence or absence of any number of occurrences of some element $x \in X$. For event-level privacy, two streams are adjacent if they differ at only one time step. Hence, in this paper, we consider event-level privacy.

2.3. Tools

In the design of differentially private mechanisms, the Laplace distribution is often used to introduce random noise [Dwork et al. 2006; Dwork 2006]. We use $\mathsf{Lap}(b)$ to denote the Laplace distribution with mean 0 and variance $2b^2$. Its probability density function is $x \mapsto \frac{1}{2b}\exp(-\frac{|x|}{b})$.

Dwork *et.al.* showed that if we mask the true answer of a query with Laplacian noise proportional to the sensitivity of the query function, such a mechanism preserves differential privacy for static databases [Dwork et al. 2006; Dwork 2006]. This is stated formally in the Fact 1.

FACT 1 (LAPLACE DISTRIBUTION MAINTAINS DIFFERENTIAL PRIVACY.). *Let $a, b \in \mathbb{R}$ and $|a - b| \leq \Delta$. Let $\gamma \sim \mathsf{Lap}(\frac{\Delta}{\epsilon})$ be a random variable having Laplace distribution. Then, for any measurable subset $S \subseteq \mathbb{R}$, $\Pr[a + \gamma \in S] \leq \exp(\epsilon) \cdot \Pr[b + \gamma \in S]$.*

In the constructions that we propose, the noise may not come from a single Laplace distribution, but rather is the sum of multiple independent Laplace distributions. We now derive a property of the sum of independent Laplace distributions.

LEMMA 2.8 (SUM OF INDEPENDENT LAPLACE DISTRIBUTIONS). *Suppose γ_i's are independent random variables, where each γ_i has Laplace distribution $\mathsf{Lap}(b_i)$. Suppose $Y := \sum_i \gamma_i$, and $b_M := \max_i b_i$. Let $\nu \geq \sqrt{\sum_i b_i^2}$ and $0 < \lambda < \frac{2\sqrt{2}\nu^2}{b_M}$. Then, $\Pr[Y > \lambda] \leq \exp(-\frac{\lambda^2}{8\nu^2})$.*

PROOF.
We use moment generating functions in a Chernoff-like argument. For each γ_i, the moment generating function is
$E[\exp(h\gamma_i)] = \frac{1}{1-h^2 b_i^2}$, where $|h| < \frac{1}{b_i}$.
Using the inequality $(1-x)^{-1} \leq 1 + 2x \leq \exp(2x)$, for $0 \leq x < \frac{1}{2}$, we have $E[\exp(h\gamma_i)] \leq \exp(2h^2 b_i^2)$, if $|h| < \frac{1}{2b_i}$.
We next use a standard calculation. For $0 < h < \frac{1}{\sqrt{2}b_M}$, we have

$$\begin{aligned}
\Pr[Y > \lambda] &= \Pr[\exp(hY) > \exp(h\lambda)] \\
&\leq \exp(-h\lambda) E[\exp(hY)] \\
&= \exp(-h\lambda) \prod_i E[\exp(h\gamma_i)] \\
&\leq \exp(-h\lambda + 2h^2\nu^2)
\end{aligned}$$

By assumption, $0 < \lambda < \frac{2\sqrt{2}\nu^2}{b_M}$. Setting $h := \frac{\lambda}{4\nu^2} < \frac{1}{\sqrt{2}b_M}$, we conclude that $\Pr[Y > \lambda] \leq \exp(-\frac{\lambda^2}{8\nu^2})$. $\square$

COROLLARY 2.9 (MEASURE CONCENTRATION). *Let Y, ν, $\{b_i\}_i$ and b_M be defined as in Lemma 2.8. Suppose $0 < \delta < 1$ and $\nu > \max\{\sqrt{\sum_i b_i^2}, b_M\sqrt{\ln \frac{2}{\delta}}\}$. Then, $\Pr[|Y| > \nu\sqrt{8 \ln \frac{2}{\delta}}] \leq \delta$.*

To simplify our presentation and improve readability, we choose $\nu := \sqrt{\sum_i b_i^2} \cdot \sqrt{\ln \frac{2}{\delta}}$ and use the following slightly weaker result: with probability at least $1 - \delta$, the quantity $|Y|$ is at most $O(\sqrt{\sum_i b_i^2} \log \frac{1}{\delta})$.

3. TIME-BOUNDED COUNTING MECHANISMS

In this section, we describe mechanisms that require a priori knowledge of an upper bound on time. In Section 4.2, we show how to remove this requirement, and achieve unbounded counting mechanisms.

3.1. Simple Counting Mechanisms

To aid the understanding of our contributions and techniques, we first explain two simple constructions.

Simple Counting Mechanism I. The mechanism is given a stream $\sigma \in \{0,1\}^{\mathbb{N}}$, a differential privacy parameter $\epsilon > 0$, and an upper bound T on time. At each time step t, the mechanism samples a fresh independent random variable $\gamma_t \sim \mathsf{Lap}(\frac{1}{\epsilon})$, and releases $\alpha_t = c(t) + \gamma_t$, where $c(t)$ is the true count at time step t. It is not hard to see that the above mechanism is $O(T\epsilon)$-differentially private, and at each time step, the error is $O(\frac{1}{\epsilon})$ with high probability. Alternatively, one can substitute $\epsilon' = \epsilon/T$, and add much bigger noise $\sim \mathsf{Lap}(\frac{1}{\epsilon'})$ at every time step. In this way, we get ϵ differential privacy; however, now the error at each time step is $O(\frac{T}{\epsilon})$.

Simple mechanism I is a straightforward extension of the Laplace mechanism proposed by Dwork *et.al.* [Dwork et al. 2006; Dwork 2006]. Basically, at every time step, the mechanism answers a new query, and randomizes the answer with fresh independent noise. The down side of this approach is that the privacy loss grows linearly with respect to the number of queries, which is t in our setting.

Simple Counting Mechanism II. In essence, Simple Counting Mechanism II produces a "sanitized" stream by adding independent Laplacian noise to each item in the stream. Suppose the mechanism is given a stream $\sigma \in \{0,1\}^{\mathbb{N}}$ and a differential privacy parameter $\epsilon > 0$. For each time step $t \in \mathbb{N}$, the mechanism samples an independent random variable γ_t with Laplace distribution $\mathsf{Lap}(\frac{1}{\epsilon})$. Define $\alpha_t := \sigma(t) + \gamma_t$. Then, the mechanism $\mathcal{M}$ gives the output $\mathcal{M}(\sigma)(t) := \sum_{i \leq t} \alpha_i$ at time t. A similar idea has been proposed as a survey technique by Warner [Warner 1965].

It is not hard to see that Simple Mechanism II can be implemented with $O(1)$ words of memory and is unbounded and ϵ-differentially private. We use Corollary 2.9 to analyze the utility of the mechanism. Fix some time T. Observe that $\mathcal{M}(\sigma)(T) - c_\sigma(T) = \sum_{t \leq T} \gamma_t =: Y$. In this case, all $\gamma_t \sim \mathsf{Lap}(\frac{1}{\epsilon})$. Hence, all $b_t := \frac{1}{\epsilon}$.

THEOREM 3.1. *Let $0 < \delta < 1$, $\epsilon > 0$. the Simple Counting Mechanism II is ϵ-differentially private, and is $(O(\frac{\sqrt{t}}{\epsilon} \cdot \log \frac{1}{\delta}), \delta)$-useful at any time $t \in \mathbb{N}$.*

PROOF. Fix $t \in \mathbb{N}$. Observe the error at time t is the sum of t independent $\mathsf{Lap}(\frac{1}{\epsilon})$ distributions. Hence, Corollary 2.9 gives the required high probability statement. □

3.2. Intuition

We will describe the Two-Level Counting Mechanism and the Binary Counting Mechanism. Informally, the Two-Level Mechanism achieves ϵ-differential privacy and $O(t^{\frac{1}{4}})$ error. The

Binary Mechanism is a further improvement, and achieves $O((\log t)^{1.5})$ error while maintaining ϵ-differential privacy. We now explain the intuitions for the Two-Level Mechanism and the Binary Mechanism.

A framework for describing mechanisms. We will describe our counting mechanisms using a common framework. Recall that the job of the mechanism is to output an approximate count at every time. However, from now on, we will think of our mechanisms as releasing noisy "p-sums" instead of counts. One can think of p-sums as intermediate results from which an observer can estimate the count at every time step herself.

Definition 3.2 (*p-sum*). A p-sum is a partial sum of consecutive items. Let $1 \leq i \leq j$. We use the notation $\Sigma[i, j] := \sum_{k=i}^{j} \sigma(k)$ to denote a partial sum involving items i through j.

Furthermore, once we add noise to a p-sum, we obtain a noisy p-sum denoted as $\widehat{\Sigma}$.

The mechanisms we consider will release noisy versions of these p-sums as new items arrive. When an observer sees the sequence of p-sums, she can compute an estimate for the count at each time step, in particular, by summing up an appropriate selection of p-sums. For example, if an observer sees a noisy p-sum $\widehat{\Sigma}[1, k] = \Sigma[1, k] + \textit{noise}$ released at time step k, and another noisy p-sum $\widehat{\Sigma}[k + 1, t] = \Sigma[k + 1, t] + \textit{noise}$ released at time step t, then she can estimate the count at time t by summing up these two noisy p-sums, i.e., $\widehat{\Sigma}[1, k] + \widehat{\Sigma}[k + 1, t]$. Notice that the observer needs to be able to do this not only for a specific time t, but also for every time step in $\mathbb{N}$.

Now we rethink Simple Mechanism I using this framework. The noisy p-sums released are noisy versions of the true count for each time step, that is, $\{\widehat{\Sigma}[1, t] = \Sigma[1, t] + \textit{noise}\}_{1 \leq t \leq T}$, where $\Sigma[1, t] = c(t)$ is the true count at time t. In this case, the $\widehat{\Sigma}[1, t]$ itself is the estimated count at time t; and therefore can be regarded as a sum of noisy p-sums (with only one summand). Notice that each item $\sigma(t)$ appears in $O(T)$ of these p-sums. This means that when you flip an item in the incoming stream, $O(T)$ of these p-sums will be affected – this is the reason why the privacy loss is linear in T.

Now consider Simple Mechanism II. The noisy p-sums released are noisy versions of each item $\widehat{\Sigma}_t = \Sigma[t, t] + \textit{noise}$, where $\Sigma[t, t] = \sigma(t)$ is the t-th item itself. In this case, each item appears in only one p-sum, however, each count is the sum of $O(T)$ p-sums. More specifically, to estimate the count at time t, the observer sums up t noisy p-sums $\widehat{\Sigma}_1, \ldots \widehat{\Sigma}_t$. As each noisy p-sum contains some fresh independent noise, the noises add up. In fact, over t time steps, the error would be $O(\sqrt{t})$ with high probability.

OBSERVATION 1. *(Informal.) Suppose a mechanism $\mathcal{M}$ adds* $\mathsf{Lap}(\frac{1}{\epsilon})$ *noise to every p-sum before releasing it. In $\mathcal{M}$, each item in the stream appears in at most x p-sums, and each estimated count is the sum of at most y p-sums. Then, the mechanism $\mathcal{M}$ achieves $x \cdot \epsilon$ differential privacy. Moreover, from Corollary 2.9, the error is $O(\frac{\sqrt{y}}{\epsilon})$ with high probability. Alternatively, to achieve ϵ-differential privacy, one can scale appropriately by having $\epsilon' = \frac{\epsilon}{x}$. Now if the mechanism instead adds* $\mathsf{Lap}(\frac{1}{\epsilon'})$ *noise to each p-sum, we achieve ϵ-differential privacy, and $O(\frac{x\sqrt{y}}{\epsilon})$ error with high probability.*

Goal. It is evident that an inherent tension exists between utility (i.e., small error) and privacy, and our challenge is how to strike a balance between the two conflicting goals. We would like to achieve the following goals.

— *Each item appears in a small number of p-sums* . Intuitively, this limits the influence of any item and guarantees small privacy loss. More specifically, when one flips an item in the incoming stream, not too many p-sums will be affected.

Table I. Informal intuition for the Two-Level Mechanism and the Binary Mechanism.

Mechanism	**Each item appears in ? p-sums**	**Each count is the sum of ? p-sums**	**Asymptotic error** (while maintaining ϵ diff. priv.)
Simple I	$O(T)$	$O(1)$	$O(T)$
Simple II	$O(1)$	$O(T)$	$O(\sqrt{T})$
Two-Level	$O(1)$	$O(\sqrt{T})$	$O(T^{\frac{1}{4}})$
Binary	$O(\log T)$	$O(\log T)$	$O((\log T)^{1.5})$

Note: For simplicity, we omit the parameters ϵ and δ from the bounds.

— *Each count is a sum of a small number of p-sums* . Each noisy p-sum contains some noise, and the noises add up as one sums up several noisy p-sums. However, if each output count is the sum of a small number of noisy p-sums, the accumulation of noises is bounded. In this way, we can achieve small error.

3.3. Two-Level Counting Mechanism

Using the Simple Counting Mechanism II as a building block, we describe the Two-Level Counting Mechanism. The idea is that when items from the stream come, we group them in contiguous blocks of size B, Within a block, we run the Simple Counting Mechanism II. On top of that, we run another Simple Counting Mechanism II, treating each block as a single element.

Algorithm 1: Two-Level Mechanism $\mathcal{D}$

```
Input: An upper bound $T$, a differential privacy parameter $\epsilon$, and a stream $\sigma \in \{0,1\}^T$
Output: At each time step $t$, output estimate $\mathcal{D}(t)$.
Initialization: Each $\alpha_i$ and $\beta_i$ are initialized to 0.
for $t \leftarrow 1$ to $T$ do
    $\alpha_t \leftarrow \sigma(t) + \mathsf{Lap}(\frac{1}{\epsilon})$
    Let $t = qB + r$ where $q, r \in \mathbb{Z}$ and $0 \leq r < B$.
    if $r = 0$ then
        $\beta_q := \sum_{i=t-B+1}^{t} \sigma(i) + \mathsf{Lap}(\frac{1}{\epsilon})$
    end
    Output
        $\mathcal{D}(t) \leftarrow \sum_{i=1}^{q} \beta_i + \sum_{i=qB+1}^{t} \alpha_i$    (1)
end
```

Two-Level Mechanism: the p-sum view. One good way to understand the Two-Level Mechanism is to consider it under the p-sum framework. Notice that in Algorithm 1, each $\beta_q = \widehat{\Sigma}[(q-1)B+1, qB]$ is a noisy p-sum, and each $\alpha_t = \widehat{\Sigma}[t,t]$ is also a noisy p-sum (for a single item). It would suffice if the mechanism simply released the set of noisy p-sums:

$$\{\beta_q | 1 \leq q \leq \lfloor T/B \rfloor\} \cup \{\alpha_t | 1 \leq t \leq T\},$$

as an observer can reconstruct the approximate count at any time step from these noisy p-sums, according to Equation (1) of Algorithm 1.

Observe that each item $\sigma(t)$ appears in at most two p-sums: at most one of the β's and at most one of the α's. Specifically, let $q := \lceil \frac{t}{B} \rceil$, then $\sigma(t)$ appears in only β_q and α_t. Hence, we can conclude that the counting mechanism preserves 2ϵ-differential privacy.

From Equation (1), it is not hard to see that the estimated count at any time t is the sum of at most $\lfloor t/B \rfloor + B$ noisy p-sums. In particular, if we let $B = \sqrt{T}$, then the estimated count at any time is the sum of at most $2B$ noisy p-sums. According to Observation 1, the error is roughly $O(T^{\frac{1}{4}}/\epsilon)$ with high probability.

We formalize the above intuition with the following theorem.

THEOREM 3.3. *The Two-Level Counting Mechanism is 2ϵ-differentially private. Furthermore, for each $t \in \mathbb{N}$, the Two-Level Counting Mechanism with block size B is $(O(\frac{1}{\epsilon} \cdot \sqrt{(\frac{t}{B} + B)} \cdot \log \frac{1}{\delta}), \delta)$-useful at time t.*

PROOF. The differential privacy argument is straightforward. As mentioned above, each item in the stream $\sigma(t)$ appears in at most 2 noisy p-sums. Therefore, if we flip $\sigma(t)$, at most 2 noisy p-sums will be affected.

We now prove the utility part of the theorem. Observe that at any time $t = qB + r$ where $q, r \in \mathbb{Z}$ and $0 \leq r < B$, the error $\mathcal{D}(t) - c_\sigma(t)$ the sum of $K = q + r$ independent Laplacian distributions $\mathsf{Lap}(\frac{1}{\epsilon})$. Since $\frac{t}{B} \leq K \leq (\frac{t}{B} + B)$, it follows as in the proof for the Simple Counting Mechanism II that, at time T, the Two-Level Counting Mechanism is $(O(\frac{1}{\epsilon} \cdot \sqrt{(\frac{t}{B} + B)} \cdot \log \frac{1}{\delta}), \delta)$-useful at time t. $\square$

Given $T \in \mathbb{N}$, we can set $B := \left\lfloor \sqrt{T} \right\rfloor$ to form a T-bounded counting mechanism.

COROLLARY 3.4. *Let $0 < \delta < 1$ and $\epsilon > 0$. For each $T \in \mathbb{N}$, there is a T-bounded counting mechanism that preserves 2ϵ-differential privacy and is $(O(\frac{1}{\epsilon} \cdot T^{1/4} \cdot \log \frac{1}{\delta}), \delta)$-useful at each time $t \in [T]$.*

Finally, we point out that the Two-Level Mechanism can actually be implemented with $O(1)$ memory.

CLAIM 1. *The Two-Level Counting Mechanism can be implemented with $O(1)$ words of memory.*

PROOF. At any time $t = qB + r$ where $q, r \in \mathbb{Z}$ and $0 \leq r < B$, the mechanism only needs to store the following values: $\sum_{i=1}^{q} \beta_i$, $\Sigma[qB + 1, t]$ and $\sum_{i=qB+1}^{t} \alpha_i$. $\square$

3.4. Binary Counting Mechanism

We could extend the idea of the Two-Level Counting Mechanism to a Multi-level Counting Mechanism, and compute the optimal number of levels given T, the upper bound on time. However, we take a better approach called the Binary Mechanism. The idea is that at any time t, the counting mechanism internally groups the items that have arrived to form p-sums of different sizes. The precise grouping of the items depends on the binary representation of the number t – hence the name Binary Mechanism.

Given any number $t \in N$, let $\mathsf{Bin}_i(t) \in \{0, 1\}$ be the ith digit in the binary representation of t, where $\mathsf{Bin}_0(t)$ is the least significant digit. Hence, $t = \sum_i \mathsf{Bin}_i(t) \cdot 2^i$. Informally, if $\mathsf{Bin}_i(t) = 1$, then there is a p-sum involving 2^i items. We formally describe the Binary Mechanism in Algorithm 2.

Binary mechanism: the p-sum view. The best way to understand the Binary Mechanism is to think in terms of the p-sum framework described earlier. Basically, instead of outputting the estimated counts, the mechanism could equivalently release a sequence of noisy p-sums which provide sufficient information for an observer to estimate the count at each time step t.

The intuitiion is best explained using a binary interval tree as shown in Figure 1. Each leaf node in the tree represents a time step, and each interior node represents a range. Intuitively, we "release" a p-sum corresponding to each node in the tree. To recover the sum of time steps 1 through t, it suffices to find a set of nodes in the tree to uniquely cover the range $[1, t]$. It is not hard to see that 1) every time step appears in only $O(\log T)$ p-sums; and 2) every contiguous range $[1, t]$ can be represented with a set of $O(\log T)$ nodes in the tree.

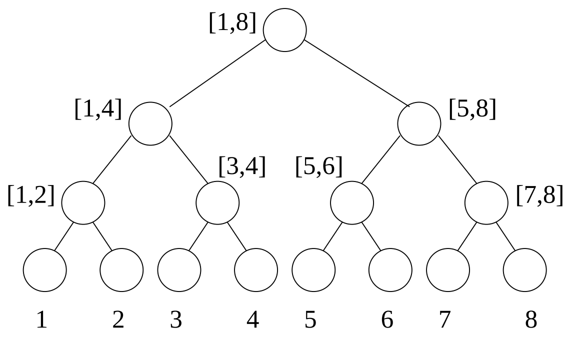
Algorithm 2: Binary Mechanism $\mathcal{B}$

```
Input: A time upper bound $T$, a privacy parameter $\epsilon$, and a stream $\sigma \in \{0,1\}^T$.
Output: At each time step $t$, output estimate $\mathcal{B}(t)$.
Initialization: Each $\alpha_i$ and $\widehat{\alpha_i}$ are (implicitly) initialized to 0.
$\epsilon' \leftarrow \epsilon / \log T$
for $t \leftarrow 1$ to $T$ do
    Express $t$ in binary form: $t = \sum_j \mathsf{Bin}_j(t) \cdot 2^j$.
    Let $i := \min\{j : \mathsf{Bin}_j(t) \neq 0\}$.
        $\alpha_i \leftarrow \sum_{j<i} \alpha_j + \sigma(t)$    (2)
    // previous value (if any) of $\alpha_i$ is overwritten
    // $\alpha_i = \Sigma[t - 2^i + 1, t]$ is a p-sum of involving $2^i$ items
    for $j \leftarrow 0$ to $i - 1$ do
        $\alpha_j \leftarrow 0, \quad \widehat{\alpha_j} \leftarrow 0$
    end
        $\widehat{\alpha_i} \leftarrow \alpha_i + \mathsf{Lap}(\frac{1}{\epsilon'})$    (3)
    // $\widehat{\alpha_i}$ is the noisy p-sum $\widehat{\Sigma}[t - 2^i + 1, t]$
    Output the estimate at time $t$:
        $\mathcal{B}(t) \leftarrow \sum_{j:\mathsf{Bin}_j(t)=1} \widehat{\alpha_j}$    (4)
end
```

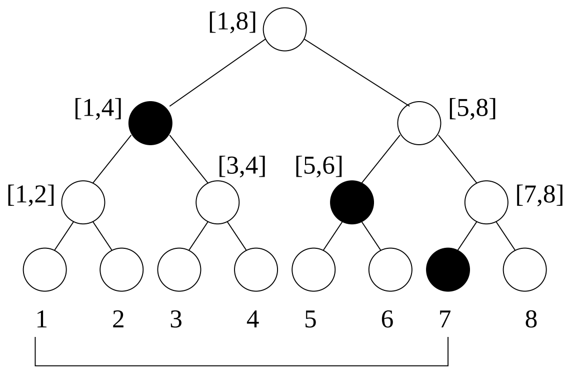
Implementing the Binary Mechanism with small memory. Observe that the mechanism only needs to store p-sums required for estimating the count at a future point in time, henceforth referred to as *active* p-sums. The mechanism can safely discard p-sums that are no longer needed, henceforth referred to as *inactive* p-sums. For example, at $t = 2^k$ where $k \in \mathbb{N}$, the only active p-sum is $\Sigma[1,t]$. All other p-sums computed between time 1 and t will no longer be needed after time $t = 2^k$. In the Binary Mechanism, we save the active p-sums in variables α_j's. We reuse these variables, and let new and active p-sums overwrite old and inactive ones. As a result, we only need to track $O(\log t)$ p-sums and their noisy versions at time t.

To be more concrete, we describe what happens when a new item arrives at time t by making an analogy to a binary counter incremented from $t-1$ to t. Let i be the position of the least significant non-zero bit in t. Then $t-1$ has $i-1$ trailing 1's in its binary representation, and a carry occurs at position i when $t-1$ is incremented by 1. At the end of time $t-1$, the mechanism stores p-sums α_j of sizes 2^j for each $j < i$. During time step t, the mechanism performs the update $\alpha_i \leftarrow \sum_{j<i} \alpha_j + \sigma(t)$. Now α_i is a new p-sum of the most recent 2^i items ending at time t. The mechanism adds noise to α_i with fresh randomness, and stores the corresponding noisy version $\widehat{\alpha}_i$. Since the p-sums α_j for $j < i$ are no longer needed, they (together with their noisy versions) are set to 0.

Differential Privacy. Consider an item arriving at $t \in [T]$. We analyze which of the p-sums would be affected if $\sigma(t)$ is flipped. It is not hard to see that the item $\sigma(t)$ can be in at most $\log T$ p-sums. In particular, it can be in at most 1 p-sum of size 2^j, where $j \leq \log T$.



(a) Every node in the binary interval tree corresponds to a p-sum.



(b) The sum of time steps 1 through 7 can be obtained by adding the p-sums corresponding to the black nodes.

Fig. 1. Intuition for the Binary Mechanism.

Observe that each noisy p-sum maintains $\frac{\epsilon}{\log T}$-differential privacy by Fact 1. Hence, we can conclude the ϵ-differential privacy of the Binary Mechanism.

THEOREM 3.5 (DIFFERENTIAL PRIVACY). *For $T \in \mathbb{N}$, the Binary Mechanism preserves T-bounded ϵ-differential privacy.*

Utility. We next consider the usefulness of the Binary Mechanism. Each estimated count $\mathcal{B}(t)$ is the sum of at most $\log T$ noisy p-sums, and each noisy p-sum contains fresh, independent Laplace noise $\mathsf{Lap}(\frac{\log T}{\epsilon})$. Therefore, the error at time t is the summation of at most $O(\log t)$ i.i.d. Laplace distributions $\mathsf{Lap}(\frac{\log T}{\epsilon})$. We use the Corollary 2.9 to conclude the mechanism's usefulness.

THEOREM 3.6 (UTILITY). *For each $t \in [T]$, the T-bounded Binary Mechanism is $(O(\frac{1}{\epsilon}) \cdot (\log T) \cdot \sqrt{\log t} \cdot \log \frac{1}{\delta}, \delta)$-useful at time $t \in [T]$.*

4. UNBOUNDED COUNTING MECHANISMS

Previously, we mainly considered time-bounded mechanisms, i.e., the mechanism requires a priori knowledge of an upper bound on the time. We now describe how to remove this assumption and derive unbounded counting mechanisms from the Binary Mechanism. The first approach, referred to as the modified Binary Mechanism, adds varying noise to each p-sum.

The second approach, referred to as the Hybrid Mechanism, gives a generic way to convert any time-bounded mechanism $\mathcal{M}$ into an unbounded one by running two mechanisms in parallel: (1) an unbounded mechanism that only outputs counts at time steps t being powers of 2; (2) a time-bounded mechanism $\mathcal{M}$ to take care of items arriving at time steps between successive powers of 2.

4.1. Modified Binary Mechanism

The time-bounded Binary Mechanism needs to know T, an upper bound on time, in order to decide the magnitude of noise added to each p-sum. How do we decide an appropriate magnitude of noise to add without knowledge of T? One idea is to add varying noise to each p-sum, depending on the length of the p-sum.

The modified Binary Mechanism. The modified Binary Mechanism works the same way as the Binary Mechanism, except for the magnitude of noise we add to each p-sum. Suppose we add $\mathsf{Lap}(\frac{a_i}{\epsilon})$ noise to a p-sum consisting of 2^i elements, where $a_i > 0$. In other words, replace Equation (3) in Algorithm 2 with the following:

$$\widehat{\alpha_i} \leftarrow \alpha_i + \mathsf{Lap}(\frac{a_i}{\epsilon})$$

We leave the a_i's as undetermined parameters for now, and work out the suitable values below.

THEOREM 4.1 (DIFFERENTIAL PRIVACY). *For $T \in \mathbb{N}$, the modified Binary Mechanism preserves T-bounded $(\epsilon \sum_{i \leq K} \frac{1}{a_i})$-differential privacy, where $K = \log_2 T + O(1)$. Moreover, if the series $\sum_i \frac{1}{a_i}$ converges to some value $A < \infty$, then the modified Binary Mechanism preserves ϵA-differential privacy.*

THEOREM 4.2 (UTILITY). *The modified Binary Mechanism is $(O(\frac{1}{\epsilon} \cdot \sqrt{\sum_{i=0}^{k} a_i^2} \cdot \log \frac{1}{\delta}), \delta)$-useful at time t, where $k := \max\{i : \mathsf{Bin}_i(t) = 1\} = \log_2 t + O(1)$.*

Remark 4.3. Using Theorems 4.1 and 4.2, and scaling ϵ, we can conclude that there is a T-bounded counting mechanism that preserves T-bounded ϵ-differential privacy, and is $(O(\frac{1}{\epsilon} \cdot \log \frac{1}{\delta}) \cdot \sum_{i=0}^{K} \frac{1}{a_i} \cdot \sqrt{\sum_{i=0}^{K} a_i^2}, \delta)$-useful at time T, where $K := \log_2 T + O(1)$. Hence, loosely speaking, with fixed privacy parameter ϵ, the error is $\sum_{i=0}^{K} \frac{1}{a_i} \cdot \sqrt{\sum_{i=0}^{K} a_i^2}$, and the best choice of a_i's should minimize the above error term. In fact, the error term $\sum_{i=0}^{K} \frac{1}{a_i} \cdot \sqrt{\sum_{i=0}^{K} a_i^2}$ is minimized when all a_i's are equal, and this is exactly our time-bounded Binary Mechanism, where $a_i = \log T$ for all $1 \leq i \leq K$.

We next give different choices of a_i's to obtain unbounded counting mechanisms with different guarantees.

COROLLARY 4.4. *Let $0 < \delta < 1$, $\epsilon > 0$. Suppose $\theta > 0$, and set $a_i := (i+1)^{1+\theta}$. There is an unbounded counting mechanism that preserves ϵ-differential privacy and is $(O(\frac{1}{\theta\epsilon} \cdot (\log t)^{1.5+\theta} \cdot \log \frac{1}{\delta}, \delta)$-useful at time t.*

In Corollary 4.4, if we choose $\theta > 0$ to be arbitrarily small, there will be a factor of $\frac{1}{\theta}$ in the error. Instead, we choose a_i to be a function that is slightly super linear in i.

For $n \in \mathbb{N}$, define $\kappa(n) := \max\{r \geq 0 : \ln^{(r)} n \geq 1\} = \Theta(\log^* n)$. Recall that $\log^*$ is defined in terms of base 2 logarithm.

For $n \in \mathbb{N}$, define the function $\mathsf{Ln}(n) := \prod_{r=0}^{\kappa(n)} \ln^{(r)} n = n(\ln n)(\ln\ln n)(\ln\ln\ln n)\cdots$.

CLAIM 2. *Let $K \in \mathbb{N}$. Then, $\sum_{i=1}^{K} \frac{1}{\mathsf{Ln}(i)} \leq \Theta(\log^* K)$ and $\sqrt{\sum_{i=1}^{K} \mathsf{Ln}(i)^2} \leq O(K^{1.5}\mathsf{Ln}(\log K))$.*

PROOF. The second statement is trivial, since for $K \geq 3$, $\sqrt{\sum_{i=1}^{K} \mathsf{Ln}(i)^2} \leq \sqrt{K\mathsf{Ln}(K)^2} = K^{1.5}\mathsf{Ln}(\ln K)$.

For $r \geq 0$, define $S(r) := \{i \in \mathbb{N} : \kappa(i) = r\}$. Then, $\sum_{i=1}^{K} \frac{1}{\mathsf{Ln}(i)} \leq \sum_{r=0}^{\kappa(K)} \sum_{i \in S(r)} \frac{1}{\mathsf{Ln}(i)}$.

We next show that for each $r \geq 0$, $\sum_{i \in S(r)} \frac{1}{\mathsf{Ln}(i)} \leq 2$. Observe this implies immediately that $\sum_{i=1}^{K} \frac{1}{\mathsf{Ln}(i)} \leq 2\kappa(K) = \Theta(\log^* K)$.

Define $a := \min S(r)$ and $b := \max S(r)$. Observe that $\mathsf{Ln}(a) \geq 1$. Hence, if $a = b$, then $\sum_{i \in S(r)} \frac{1}{\mathsf{Ln}(i)} = \frac{1}{\mathsf{Ln}(a)} \leq 1 < 2$. We assume $a + 1 \leq b$.

Define for $x \in [a, b]$, the function $f(x) := \prod_{j=0}^{r} \ln^{(j)} x$. Observe that for all $i \in S(r)$, $\mathsf{Ln}(i) = f(i)$. Moreover, f is monotonically increasing, and we have the indefinite integral $\int \frac{dx}{f(x)} = \ln^{(r+1)} x + C$. Therefore, we have

$\sum_{i \in S(r)} \frac{1}{\mathsf{Ln}(i)} = \frac{1}{\mathsf{Ln}(a)} + \sum_{i=a+1}^{b} \frac{1}{f(i)} \leq 1 + \int_a^b \frac{dx}{f(x)} = 1 + \ln^{(r+1)} b - \ln^{(r+1)} a \leq 2,$

where the last inequality holds because for all $i \in S(r)$, $0 \leq \ln^{(r+1)} i < 1$. $\square$

COROLLARY 4.5. *Let $0 < \delta < 1$, $\epsilon > 0$. Set $a_i := \mathsf{Ln}(i+1)$. There is an unbounded counting mechanism, such that for any $t \in \mathbb{N}$, it preserves $O(\epsilon \log^*(\log t))$-differential privacy, and is $(O(\frac{1}{\epsilon}) \cdot \log^{1.5} t \cdot \mathsf{Ln}(\log\log t) \cdot \log\frac{1}{\delta}, \delta)$-useful at time t.*

4.2. Hybrid Mechanism

The modified Binary Mechanism is unbounded, and achieves guarantees similar to the Binary Mechanism, but with a slight penalty in the error term (under fixed privacy parameter ϵ). We now describe a better approach that achieves time unboundedness and meanwhile provides the same asympototic bounds as the Binary Mechanism.

The idea is to have an unbounded mechanism which only reports the estimated counts at sparse intervals, in particular, when t is a power of 2. We would expect such a mechanism to have better guarantees than one that has to report at every time step.

So what do we do when t is not a power of 2? We know the approximate count $\widehat{c}_1$ for the time period $[1, T]$ where $T = 2^k$ for some non-negative integer k. Suppose we also know the approximate count $\widehat{c}_2$ for the time period $[T + 1, t]$ where $T + 1 \leq t \leq 2T$. Then we can estimate the count at time t as $\widehat{c}_1 + \widehat{c}_2$. Therefore, it remains for us to count the 1's between $[T, t]$ for any $t \in [T + 1, 2T]$, We can simply apply a T-bounded mechanism (e.g., the Binary Mechanism) for this task.

Logarithmic Counting Mechanism. We now design an unbounded mechanism called the *Logarithmic Mechanism* which reports the count only when the time t is a power of 2.

Algorithm 3: Logarithmic Mechanism $\mathcal{L}$

```
Input: Differential privacy parameter $\epsilon$, and a stream $\sigma \in \{0,1\}^{\mathbb{N}}$.
Output: $\forall k \in \mathbb{Z}$, at time $t = 2^k$, output estimate $\mathcal{L}(t)$.
Initialization: $\beta \leftarrow 0$.
foreach $t \in \mathbb{N}$ do
    $\beta \leftarrow \beta + \sigma(t)$
    if $t = 2^k$ for some $k \in \mathbb{Z}$ then
        $\beta \leftarrow \beta + \mathsf{Lap}(\frac{1}{\epsilon})$
        Output $\mathcal{L}(t) \leftarrow \beta$
    end
end
```

The idea for the Logarithmic Mechanism is quite simple. The mechanism internally keeps a value β which is initialized to 0. β is used to keep track of the approximate count at any point of time. As an item comes in, its value is added to β. At t equal to a power of 2, the mechanism adds fresh randomness to the value β (on top of randomness previously added), and outputs β.

If t is a power of 2, it is clear that the accumulated error at time t is a sum of $O(\log t)$ independent Laplace distributions $\mathsf{Lap}(\frac{1}{\epsilon})$. Hence, we have the following guarantee from Corollary 2.9.

THEOREM 4.6. *The Logarithmic Counting Mechanism is unbounded, preserves ϵ-differential privacy and is $(O(\frac{1}{\epsilon}) \cdot \sqrt{\log t} \cdot \log \frac{1}{\delta}, \delta)$-useful at time $t = 2^k$ for all $k \geq 0$.*

Logarithmic Mechanism: the p-sum view. The Logarithmic Mechanism also has a p-sum interpretation. Equivalently, one can think of it as releasing the noisy p-sums $\widehat{\alpha}_0 = \widehat{\Sigma}[1,1]$, as well as $\widehat{\alpha}_k = \widehat{\Sigma}[2^{k-1}+1, 2^k]$ for every $k \geq 1$, Now an observer can estimate the count at time $t = 2^k$ as $\sum_{i=0}^{k} \widehat{\alpha}_i$.

Hybrid Mechanism. We combine the Logarithmic Mechanism and a time-bounded counting mechanism to process a given stream σ. We run one copy of $\frac{\epsilon}{2}$-differentially private Logarithmic Mechanism, which reports an approximate count when t is a power of 2. Suppose the Logarithmic Mechanism has reported count $\mathcal{L}(T)$ at $T = 2^k$ for some non-negative integer k. For time t in the range $T+1 \leq t \leq 2T$, we run an $\frac{\epsilon}{2}$-differentially private T-bounded counting mechanism denoted as $\mathcal{M}$ to count the number of 1's in the range $[T+1, t]$. We write $\tau = t - T$. At time t, let $\mathcal{M}(\tau)$ be the number of 1's in $[T+1, T+\tau]$ reported by the T-bounded counting mechanism $\mathcal{M}$. Then, the hybrid mechanism reports $\mathcal{L}(T) + \mathcal{M}(\tau)$ at time t. The detailed Hybrid Mechanism is presented in Algorithm 4.

THEOREM 4.7. *Assume that given any $\epsilon > 0$ and $0 < \delta < 1$, Logarithmic Mechanism $\mathcal{L}$ is ϵ-differentially private and is $(f(\epsilon, t, \delta), \delta)$-useful at time t. Similarly, assume that given any $\epsilon > 0$, the T-bounded mechanism $\mathcal{M}$ is ϵ-differentially private and is $(g(\epsilon, T, \tau, \delta), \delta)$-useful at time $\tau \in [T]$, where g is monotonically increasing with T and τ. Then, the Hybrid Mechanism described above is unbounded, preserves ϵ-differential privacy, and is $(f(\frac{\epsilon}{2}, t, \frac{\delta}{2}) + g(\frac{\epsilon}{2}, t, t, \frac{\delta}{2}), \delta)$-useful at time t.*

PROOF. We first note that the $\frac{\epsilon}{2}$-differentially private Logarithmic Counting Mechanism is run in parallel with at most one instance of the $\frac{\epsilon}{2}$-differentially private time bounded counting mechanism at any time. Hence, the Hybrid Counting Mechanism is ϵ-differentially private.

Suppose at time $t = 2^k$ (for some non-negative integer k), the Logarithmic Counting Mechanism has reported some count $\mathcal{L}(t)$. From the assumption, we know that with probability at least $1 - \frac{\delta}{2}$, the error of $\mathcal{H}(t) := \mathcal{L}(t)$ is at most $f(\frac{\epsilon}{2}, t, \frac{\delta}{2})$.

Algorithm 4: Hybrid Mechanism $\mathcal{H}$ (with Mechanism $\mathcal{M}$).

```
Input: Differential privacy parameter $\epsilon$, a stream $\sigma \in \{0,1\}^{\mathbb{N}}$, Logarithmic Mechanism $\mathcal{L}$, and a
       time-bounded mechanism $\mathcal{M}$.
Output: For each $t \in \mathbb{N}$, output estimate $\mathcal{H}(t)$.
Initialization: $T \leftarrow 1$.
Initiate the mechanism $\mathcal{L}$ with privacy parameter $\frac{\epsilon}{2}$ on stream $\sigma$.
foreach $t \in \mathbb{N}$ do
    Feed $\sigma(t)$ to mechanism $\mathcal{L}$.
    if $t = 2^k$ for some $k \in \mathbb{Z}$ then
        Output $\mathcal{H}(t) \leftarrow \mathcal{L}(t)$
        $T \leftarrow t$
        Initiate an instance of the $T$-bounded mechanism $\mathcal{M}_T$ with time upper bound $T$, privacy
        parameter $\frac{\epsilon}{2}$ and stream $\sigma^{(T)} \in \{0,1\}^T$, where $\sigma^{(T)}(\tau) := \sigma(\tau + T)$ for $\tau \in [1, T]$.
    else
        $\tau \leftarrow t - T$
        Feed $\sigma^{(T)}(\tau) := \sigma(t)$ to mechanism $\mathcal{M}_T$.
        Output $\mathcal{H}(t) \leftarrow \mathcal{L}(T) + \mathcal{M}_T(\tau)$
    end
end

// At time $t$, $T$ is the largest power of 2 no bigger than $t$.
// $\sigma^{(T)}$ is the sub-stream of $\sigma$ for the duration $[T+1, 2T]$.
// $\mathcal{M}_T$ is a time-bounded mechanism that runs for $[T+1, 2T]$.
```

Consider $T + 1 \leq t \leq 2T$. We write $\tau := t - T$. Suppose the T-bounded counting mechanism $\mathcal{M}_T$ reports $\mathcal{M}_T(\tau)$ at time t. From the assumption, with probability at least $1 - \frac{\delta}{2}$, $\mathcal{M}_T(\tau)$ has error at most $g(\frac{\epsilon}{2}, T, \tau, \frac{\delta}{2}) \leq g(\frac{\epsilon}{2}, t, t, \frac{\delta}{2})$.

We conclude that with probability at least $1 - \delta$, $\mathcal{H}(t) := \mathcal{L}(T) + \mathcal{M}_T(\tau)$ has error at most $f(\frac{\epsilon}{2}, t, \frac{\delta}{2}) + g(\frac{\epsilon}{2}, t, t, \frac{\delta}{2})$. $\square$

COROLLARY 4.8 (HYBRID MECHANISM). *If we instantiate the Hybrid Mechanism using the Binary Mechanism as the T-bounded mechanism, the resulting Hybrid Mechanism is unbounded, preserves ϵ-differential privacy, and is $(O(\frac{1}{\epsilon}) \cdot (\log t)^{1.5} \cdot \log \frac{1}{\delta}, \delta)$-useful at time t.*

For simplicity, in the remainder of the paper, when we refer to the Hybrid Mechanism, we mean the Hybrid Mechanism instantiated with the Binary Mechanism.

Hybrid Mechanism: the p-sum view. One can also interpret the Hybrid Mechanism naturally using the p-sum framework. Basically, one can equivalently think of the Hybrid Mechanism as releasing the union of the noisy p-sums of the Logarithmic Mechanism and the Binary Mechanism. From this set of noisy p-sums, an observer can compute the approximate count at every time step $t \in \mathbb{N}$.

5. CONSISTENCY

The mechanisms we have described so far could report count values that are not integral and hence definitely cannot correspond to any stream of 0's and 1's. One could of course round any reported value to the nearest integer. However, this does not necessarily mean that the values reported by a mechanism throughout different time steps correspond to any valid stream.

We now formally define the consistency of a continual counting mechanism, and describe an approach to transform any counting mechanism into a consistent mechanism.

Definition 5.1 (*Consistent Mechanism*). A counting mechanism $\mathcal{M}$ is consistent, if for any stream σ, for all $t \in \mathbb{N}$, $\mathcal{M}(\sigma)(t) - \mathcal{M}(\sigma)(t-1) \in \{0, 1\}$. If $\mathcal{M}$ is randomized, this means $Pr[\mathcal{M}(\sigma)(t) - \mathcal{M}(\sigma)(t-1) \in \{0,1\}] = 1$. We use the convention $\mathcal{M}(\sigma)(0) := 0$.

Consistent Rounding. We describe a straightforward way to transform any counting mechanism $\mathcal{M}$ into a consistent mechanism $\widehat{\mathcal{M}}$. For ease of presentation, we assume that $\mathcal{M}$ reports integer values (this can be done by first rounding reported values to integers, introducing error of at most $\frac{1}{2}$). The consistent rounding procedure is simple. Given any stream σ, we set $\widehat{\mathcal{M}}(\sigma)(0) := 0$, and feed σ into $\mathcal{M}$. For each $t \in \mathbb{N}$, if $\mathcal{M}(\sigma)(t) > \widehat{\mathcal{M}}(\sigma)(t-1)$, then $\widehat{\mathcal{M}}(\sigma)(t) := \widehat{\mathcal{M}}(\sigma)(t-1) + 1$, otherwise $\widehat{\mathcal{M}}(\sigma)(t) := \widehat{\mathcal{M}}(\sigma)(t-1)$.

From the construction, at each time step, the mechanism $\widehat{\mathcal{M}}$ either increases the reported value by 1 from that in the previous time step, or does not change the reported value. It is therefore consistent. Moreover, rounding is performed online. There are more sophisticated offline rounding procedures such as Pool-Adjacent-Violators Algorithm (PAVA) [Yeganova and Wilbur 2009] to achieve consistency from noisy data. However, we note that even in the offline setting, PAVA does not give better asymptotic error bounds than our simple rounding method. The next lemma shows that our consistent rounding transformation does not introduce further error.

LEMMA 5.2 (CONSISTENT TRANSFORMATION PRESERVES ERROR). *Suppose $\mathcal{M}$ is a counting mechanism that returns integer values. Suppose that an instance of $\mathcal{M}$ is run on some stream σ that produces an output $\mathcal{M}(\sigma)$ satisfying the following condition.*

— *There exists a non-negative error function $E(t)$ that is monotonically increasing in t such that for all $t \geq 0$, $|c_\sigma(t) - \mathcal{M}(\sigma)(t)| \leq E(t)$.*

Then, the above consistent rounding procedure produces an output $\widehat{\mathcal{M}}(\sigma)$ that satisfies the same error bound, i.e., for all $t \geq 0$, $|c_\sigma(t) - \widehat{\mathcal{M}}(\sigma)(t)| \leq E(t)$.

PROOF. We prove the Lemma by induction on t. Since $c_\sigma(0) = \mathcal{M}(\sigma)(0) = \widehat{\mathcal{M}}(\sigma)(0)$, we trivially have $|c_\sigma(0) - \widehat{\mathcal{M}}(\sigma)(0)| = 0 \leq E(0)$.

Assume that for some $t \geq 0$, $t \geq 0$, $|c_\sigma(t) - \widehat{\mathcal{M}}(\sigma)(t)| \leq E(t)$.

Observe that if $|c_\sigma(t+1) - \widehat{\mathcal{M}}(\sigma)(t+1)| \leq |c_\sigma(t) - \widehat{\mathcal{M}}(\sigma)(t)|$, then the result follows because $E(t) \leq E(t+1)$.

There are two cases where $|c_\sigma(t+1) - \widehat{\mathcal{M}}(\sigma)(t+1)| > |c_\sigma(t) - \widehat{\mathcal{M}}(\sigma)(t)|$.

Case 1: $\widehat{\mathcal{M}}(\sigma)(t+1) - 1 = \widehat{\mathcal{M}}(\sigma)(t) \geq c_\sigma(t) = c_\sigma(t+1)$. This implies that $\mathcal{M}(\sigma)(t+1) > \widehat{\mathcal{M}}(\sigma)(t)$. Since we assume $\mathcal{M}$ returns integer values, it follows that $\mathcal{M}(\sigma)(t+1) \geq \widehat{\mathcal{M}}(\sigma)(t+1) \geq c_\sigma(t+1)$. Hence, $|c_\sigma(t+1) - \widehat{\mathcal{M}}(\sigma)(t+1)| \leq |c_\sigma(t+1) - \mathcal{M}(\sigma)(t+1)| \leq E(t+1)$.

Case 2: $\widehat{\mathcal{M}}(\sigma)(t+1) = \widehat{\mathcal{M}}(\sigma)(t) \leq c_\sigma(t) = c_\sigma(t+1) - 1$. This implies that $\mathcal{M}(\sigma)(t+1) \leq \widehat{\mathcal{M}}(\sigma)(t) = \widehat{\mathcal{M}}(\sigma)(t+1) < c_\sigma(t+1)$. Hence, we also have $|c_\sigma(t+1) - \widehat{\mathcal{M}}(\sigma)(t+1)| \leq |c_\sigma(t+1) - \mathcal{M}(\sigma)(t+1)| \leq E(t+1)$.

This completes the inductive step. □

In order to apply Lemma 5.2, we need the output $\mathcal{M}(\sigma)$ to satisfy the error bound for all values of t. However, Corollary 4.8 only gives a high-probability statement for a single value of t. This can be resolved by a simple application of union bound. For each t, we use failure probability $\delta_t := \Theta(\frac{\delta}{t^2})$. Observing that $\sum_{t \in \mathbb{N}} \delta_t = \Theta(\delta)$ and $O(\log \frac{1}{\delta_t}) = O(\log t + \log \frac{1}{\delta})$, we have the following corollary.

COROLLARY 5.3. *The Hybrid Mechanism $\mathcal{M}$ preserves ϵ-differential privacy, and for any $0 < \delta < 1$, for any stream σ, with probability at least $1 - \delta$, the following error bound holds: for each $t \in \mathbb{N}$, $|c_\sigma(t) - \mathcal{M}(\sigma)(t)| \leq O(\frac{1}{\epsilon}) \cdot (\log t)^{2.5} \cdot \log \frac{1}{\delta}$.*

Hence, Lemma 5.2 and Corollary 5.3 show that the rounding procedure gives a consistent counting mechanism $\widehat{\mathcal{M}}$ with the same error bound.

6. ACHIEVING PAN PRIVACY

The mechanisms described thus far are not designed to resist intrusions in which the adversary can learn snapshots of the mechanism's internal states. We now consider how to add pan privacy, i.e., the ability to resist intrusions into the mechanisms.

6.1. Pan Privacy Definitions

Dwork *et.al.* first formalized the notion of pan privacy [Dwork et al. 2010; Dwork 2010a] to deal with intruders who can observe snapshots of the mechanism's internal states, e.g., in a subpoena. We have to assume some notion of "atomicity", that is, an intrusion can happen only when the mechanism has finished its update at a certain time step. During the update, the mechanism has to store the true value of new item $\sigma(t)$ somehow, and intrusion at this point would clearly break the privacy.

Definition 6.1 (*Pan Privacy against Single Unannounced Intrusion*). Suppose I is the set of the internal states of a mechanism $\mathcal{M}$. Given a stream σ and an unannounced intrusion at time t, we can view the output $\mathcal{M}(\sigma)$ of the mechanism as an element $(i_t, s) \in I \times \mathbb{R}^{\mathbb{N}}$, where i_t represents the knowledge gained in the one single intrusion at time t. A counting mechanism $\mathcal{M}$ is ϵ-pan private (or preserves ϵ-pan privacy) against single intrusion if for any adjacent streams σ and σ', any time t and any measurable subset $S \subseteq I \times \mathbb{R}^{\mathbb{N}}$, $\Pr[\mathcal{M}(\sigma) \in S] \leq \exp(\epsilon) \cdot \Pr[\mathcal{M}(\sigma') \in S]$.

Definition 6.2 (*Pan Privacy against Multiple Announced Intrusions*). Suppose I is the set of the internal states of a mechanism $\mathcal{M}$. Let $K \subset \mathbb{N}$ be a subset of size k that represents the time steps at which intrusions are made. We assume that an intrusion is not known in advance, but the mechanism is aware of an intrusion immediately after it has happened. Given a stream σ and the intrusion times K, we can view the output $\mathcal{M}(\sigma, K)$ as an element $(i, s) \in I^k \times \mathbb{R}^{\mathbb{N}}$, where i represents the knowledge gained in the k intrusions. A counting mechanism $\mathcal{M}$ is ϵ-pan private (or preserves ϵ-pan privacy) against multiple intrusions if for any adjacent streams σ and σ', any $K \subset \mathbb{N}$, and any measurable subset $S \subseteq I^{|K|} \times \mathbb{R}^{\mathbb{N}}$, $\Pr[\mathcal{M}(\sigma, K) \in S] \leq \exp(\epsilon) \cdot \Pr[\mathcal{M}(\sigma', K) \in S]$.

6.2. Pan Privacy Against Single Unannounced Intrusion

Recall that all our mechanisms fall within the p-sum framework described in Section 3.2. We now describe a generic technique for transforming any ϵ-differentially private mechanism in the p-sum framework into an ϵ-pan private mechanism resilient against a single unannounced intrusion, with only a constant-factor loss in the mechanism's utility. The techniques used here are similar to those proposed by Dwork *et.al.* [Dwork et al. 2010; Dwork 2010a].

A mechanism in the p-sum framework releases a sequence of noisy p-sums from which an observer can estimate the count at each time step. One way to implement such a mechanism is described in Algorithm 5.

Algorithm 5: One way to implement any mechanism in the p-sum framework.

```
Let $P$ denote the (possibly infinite) set of all noisy p-sums the mechanism intends to output.
Initialize: $S \leftarrow \emptyset$.
// $S$ is the set of currently active p-sums
foreach $t \in \mathbb{N}$ do
    // Initialize a counter for each
    // relevant p-sum starting at $t$:
    foreach noisy p-sum $\widehat{\Sigma}[t, t'] \in P$ do
        $\beta_{t,t'} \leftarrow 0$        (5)
        $S \leftarrow S \cup \{\beta_{t,t'}\}$
    end
    foreach $\beta_{t_1,t_2} \in S$ do
        // Add the current item to all relevant counters
        if $t_1 \le t \le t_2$ then
            $\beta_{t_1,t_2} \leftarrow \beta_{t_1,t_2} + \sigma(t)$
        end
        // Output noisy p-sums ending at $t$
        // and remove these counters from memory
        if $t = t_2$ then
            Output $\widehat{\Sigma}[t_1, t_2] = \beta_{t_1,t_2} + \mathsf{Lap}(\frac{a}{\epsilon})$ with appropriate choice of $a$
            $S \leftarrow S \backslash \{\beta_{t_1,t_2}\}$
        end
    end
end
```

Basically, if $\widehat{\Sigma}[t_1, t_2]$ is a noisy p-sum the mechanism intends to release, then the mechanism initializes a corresponding counter β_{t_1,t_2} at time t_1. For $t_1 \le t \le t_2$, $\beta_{t_1,t_2} = \Sigma[t_1, t]$ at the end of t. In other words, the mechanism internally keeps track of the accurate count for the duration $[t_1, t]$. At the end of time t_2, $\beta_{t_1,t_2} = \Sigma[t_1, t_2]$. Now the mechanism adds noise to the counter, and reveals the noisy p-sum $\widehat{\Sigma}[t_1, t_2]$.

Remark 6.3. If we implement the Binary Mechanism or the Hybrid Mechanism using the above approach, it is not hard to show that only $O(\log t)$ counters are needed at time t. In other words, at any time t, $|S| < O(\log t)$.

We now examine how our current mechanisms fail to achieve pan privacy, and show how to remedy the problem. Notice that if an intrusion happens at time $t_1 \le t < t_2$, then the adversary can learn $\Sigma[t_1, t]$ and privacy is obviously broken. The way to resolve this is to initialize the counter with some noise $\mathsf{Lap}(\frac{1}{\epsilon})$. Basically, replace Equation (5) with the following for an appropriate choice of a.

$$\beta_{t,t'} \leftarrow \mathsf{Lap}(\frac{a}{\epsilon})$$

Specifically, we choose the magnitude of noise a as below. Suppose the original mechanism intended to add $\mathsf{Lap}(\frac{a(t_1,t_2)}{\epsilon})$ to the p-sum $\Sigma[t_1, t_2]$. Then we initialize the counter β_{t_1,t_2} with noise $\mathsf{Lap}(\frac{a(t_1,t_2)}{\epsilon})$.

THEOREM 6.4 (PAN-PRIVACY AGAINST SINGLE UNANNOUNCED INTRUSION). *The above procedure can convert all counting mechanisms which fall within the p-sum framework into pan private versions with the same privacy guarantee and the same asymptotic error guarantee.*

PROOF. The above procedure achieves ϵ-pan privacy due to the following observations. Suppose the intrusion happens at time $t_1 \leq t \leq t_2$, the adversary learns the internal state $\beta_{t_1,t_2} = \Sigma[t_1, t] + \mathsf{Lap}(\frac{a}{\epsilon})$ in addition to the noisy p-sum $\widehat{\Sigma}[t_1, t_2] = \Sigma[t_1, t_2] + \mathsf{Lap}(\frac{a}{\epsilon}) + \mathsf{Lap}(\frac{a}{\epsilon})$ output at time t_2. This is equivalent to revealing the noisy p-sums $\widehat{\Sigma}[t_1, t] := \Sigma[t_1, t] + \mathsf{Lap}(\frac{a}{\epsilon})$ and $\widehat{\Sigma}[t+1, t_2] := \Sigma[t+1, t_2] + \mathsf{Lap}(\frac{a}{\epsilon})$ to the adversary. Note that changing any single position in the stream can only affect at most one of $\widehat{\Sigma}[t_1, t]$ and $\widehat{\Sigma}[t+1, t_2]$. Hence, this mechanism achieves ϵ-pan privacy if its non-pan private version achieves ϵ-differential privacy.

As a result of the above modification, each noisy p-sum output by the mechanism now has two independent Laplace noises, one added at the time the counter was initialized, the other added at the time of output. Hence, from Corollary 2.9, this would only lead to a constant factor $\sqrt{2}$ increase in the error bound. □

6.3. Pan Privacy Against Multiple Announced Intrusions

We can apply the same idea for multiple intrusions, as long as the mechanism is aware of the intrusions immediately after they are made. We initialize each counter β_{t_1,t_2} with fresh Laplace noise $\mathsf{Lap}(\frac{a}{\epsilon})$. Whenever we detect an intrusion, we add fresh Laplace noise $\mathsf{Lap}(\frac{a}{\epsilon})$ to each active counter in memory.

THEOREM 6.5. *If we modify the Hybrid Mechanism using the above approach, the resulting mechanism is unbounded, preserves ϵ-pan privacy against multiple intrusions, and is $(O(\frac{\sqrt{k+1}}{\epsilon}) \cdot (\log t)^{1.5} \cdot \log \frac{1}{\delta}, \delta)$-useful at time t, where k is the number of intrusions made before time t.*

PROOF. Observe that k intrusions means that a p-sum breaks into $k+1$ p-sums, each having the same privacy guarantee ϵ as before. Hence, ϵ-pan privacy is achieved.

Moreover, k additional copies of independent $\mathsf{Lap}(\frac{a}{\epsilon})$ are added to the original p-sum. Hence, from Corollary 2.9, there is a factor $\sqrt{k+1}$ associated with the error. □

7. APPLICATIONS

The counter is a fundamental primitive in many streaming algorithms [Metwally et al. 2005; Demaine et al. 2002; Manku and Motwani 2002]. We believe that our differentially private continual counter construction will inspire the design of a wide class of differentially private streaming algorithms.

7.1. Recommendation System

Our counting mechanism also has immediate practical applications. For example, it is a common practice for websites such as IMDB, Delicious and Digg to suggest the most popular movies, bookmarks or news items to visitors. Our construction can be extended to suggest top-k items or hot items in a differentially private manner.

Continual top-k suggestions. Let $\mathcal{U} := \{1, 2, \ldots, m\}$ denote the universe of possible items. Let $\sigma \in \mathcal{U}^{\mathbb{N}}$ denote a stream of incoming items. Specifically, $\sigma(t) \in \mathcal{U}$ denotes the item at time $t \in \mathbb{N}$. At time step $t \in N$, the top-k items are the k most frequent items that appear in the stream σ in the first t time steps.

At every time step $t \in \mathbb{N}$, the continual top-k mechanism $\mathcal{K}$ outputs a tuple $\mathcal{K}(\sigma)(t) \in \mathcal{U}^k$, an approximation of the top-k items seen thus far.

We say that two streams σ, σ' are adjacent if they differ at exactly one time step $t \in \mathbb{N}$. A continual top-k mechanism is ϵ-differentially private if for any adjacent streams σ and σ', and any measurable subset $S \subseteq (\mathcal{U}^k)^{\mathbb{N}}$, $\Pr[\mathcal{K}(\sigma) \in S] \leq \exp(\epsilon) \cdot \Pr[\mathcal{K}(\sigma') \in S]$.

We can use the continual counter construction to give continual top-k suggestions in a differentially private manner. To achieve this, the mechanism $\mathcal{K}$ internally runs m differentially private continual counters, to keep track of the approximate count of each item thus far. We assume that each continual counter is instantiated using the Hybrid Mechanism described in Section 4.2. At each time step, the mechanism $\mathcal{K}$ ranks the items according to their approximate counts as suggested by the m counters, and outputs the resulting top-k.

More formally, we "split" the stream σ into m different indicator streams σ_1, σ_2, …, σ_m. If $\sigma(t) = i \in \mathcal{U}$, then $\sigma_i(t) := 1$ and $\sigma_j(t) := 0$ for all $j \neq i$. The mechanism $\mathcal{K}$ instantiates m ϵ-differentially private continual counters $\mathcal{H}_1, \ldots, \mathcal{H}_m$ using the Hybrid Mechanism, and feeds the indicator stream σ_i to the counter $\mathcal{H}_i$ for each $i \in \mathcal{U}$. At time t, the mechanism $\mathcal{K}$ computes the ranking according to the approximate counts $\{\mathcal{H}_1(t), \mathcal{H}_2(t), \ldots, \mathcal{H}_m(t)\}$, and outputs the resulting top-k items.

The mechanism $\mathcal{K}$ described above preserves 2ϵ-differential privacy, and achieves good accuracy in the following sense.

CLAIM 3. *The above continual top-k mechanism $\mathcal{K}$ is 2ϵ-differentially private. Furthermore, at any time $t \in \mathbb{N}$, for any two items $i, j \in \mathcal{U}$ if the true counts of items i and j differ by $\Omega(\frac{1}{\epsilon} \cdot (\log t)^{1.5} \cdot \log \frac{1}{\delta})$, then with probability $1 - \delta$, the ranking computed by mechanism $\mathcal{K}$ preserves the correct ordering of i and j at time t.*

PROOF. For 2ϵ differential privacy, observe that flipping one item in the original stream σ affects two of the counters among $\mathcal{H}_1, \ldots, \mathcal{H}_m$. The usefulness argument follows directly from Corollary 4.8. □

Continual hot items suggestions. Hot items can be considered as a variation of top-k queries. Using a similar idea, we can design a mechanism that continually outputs hot items suggestions in a differentially private manner. Basically, we run m differentially private continual counters to keep track of the approximate count for each item, and compute the hot items list according to their approximate counts. In this way, we can achieve the same guarantees as suggested by Claim 3.

Finally, note that the mechanisms described in this section can be augmented to achieve pan privacy using techniques described in Section 6.

7.2. Multi-dimensional Range Query

We can generalize our techniques to multi-dimensional range query. Multi-dimensional range queries are widely-used in practice. In particular, database SQL queries are often by nature multi-dimensional range queries.

Example. Consider a medical database consisting of demographics information such as age and salary, and whether the patient is diabetic. Suppose medical researchers would like study how demographics influence the chance of having diabetes. Therefore, the database allows medical researchers to make multi-dimensional range queries such as:

$$\textsf{select count}(*)\ \textsf{where age} > 50\ \textsf{and salary}\ > 10K\ \textsf{and diabetic}\ = \textsf{true}$$

When the database is evolving, e.g., entries are being collected and added to the database over the course of time, time can also be considered as a special dimension. For example, at time t, if a medical research wishes to make a query over all existing records, we can essentially express the query as below:

$$\textsf{select count}(*)\ \textsf{where age} > 50\ \textsf{and salary}\ > 10K\ \textsf{and diabetic}\ = \textsf{true and time\_added}\ \leq t$$

As medical information is privacy sensitive, we wish to support such analytics without harming each individual's privacy. Our mechanism described below allows the medical researchers to make an arbitrarily large number of queries, such that the error associated with each query is independent of the number of queries.

Generalized definitions for multi-dimensional range query. In general, we formulate the problem as below. Suppose we have a d-dimensional database $\sigma : [T]^d \rightarrow \{0, 1\}$, where the outcome bit represents a predicate of interest, e.g., whether the patient is diabetic.

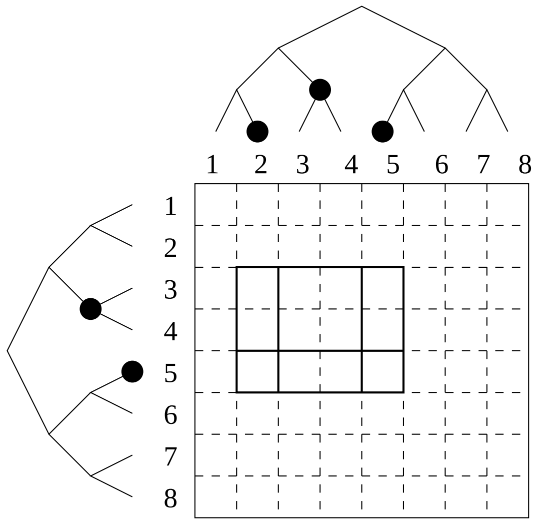


Fig. 2. Intuition for multi-dimensional range query.

Below, we treat the database as a static one, although the algorithm also readily applies to evolving databases, as we can think of time as a special dimension.

We wish to support range query of the following form. Consider the set of hyper-rectangles. Formally, $\mathcal{B} := \{I_1 \times I_2 \times \cdots \times I_d \subseteq [T]^d : \forall i \in [d], \exists 1 \leq s_i \leq t_i \leq T, I_i = [s_i, t_i]\}$. For each $B \in \mathcal{B}$, we would like an estimate of $c_\sigma(B) := \sum_{x \in B} \sigma(x)$.

Given a database σ, a mechanism $\mathcal{M}$ returns $\mathcal{M}(\sigma) : \mathcal{B} \to \mathbb{R}$, which can also be viewed as a point in $\mathbb{R}^{\mathcal{B}}$. Two databases σ and σ' are neighbors if they differ at exactly one point $x \in [T]^d$.

As before, a mechanism $\mathcal{M}$ is ϵ-differentially private if for any two neighboring databases σ and σ', any measurable $S \subseteq \mathbb{R}^{\mathcal{B}}$, $Pr[\mathcal{M}(\sigma) \in S] \leq \exp(\epsilon) \cdot Pr[\mathcal{M}(\sigma') \in S]$. Similarly, a (randomized) mechanism $\mathcal{M}$ is (λ, δ)-useful for query $B \in \mathcal{B}$ if with probability at least $1 - \delta$, $|\mathcal{M}(\sigma)(B) - c_\sigma(B)| \leq \lambda$.

Construction for multi-dimensional range query. We use the same technique as for the Binary Mechanism in Section 3.4. Recall that for the one-dimensional case, we release a series of p-sums corresponding to all nodes in the binary interval tree depicted in Figure 1. For each dimension i, let $\mathcal{I}_i$ be the set of intervals in $[1, T]$ corresponding to the p-sums used in the Binary Mechanism, i.e., nodes in the binary interval tree as in Figure 1. In particular, if T is a power of 2, then there are exactly $\frac{T}{2^j}$ intervals of length 2^j. In general, $|\mathcal{I}_i| = O(T)$.

Let $\mathcal{I} := \mathcal{I}_1 \times \mathcal{I}_2 \times \cdots \times \mathcal{I}_d$. Each $(I_1, I_2, \ldots, I_d) \in \mathcal{I}$ can be identified with the rectangle $I_1 \times I_2 \times \cdots \times I_d \in \mathcal{B}$, hencethforth referred to as a *basic rectangle*. The idea is that for each basic rectangle $B \in \mathcal{I}$, the mechanism computes $c_\sigma(B)$ and releases a noisy p-sum $\widehat{\Sigma}(B) := c_\sigma(B) + \mathsf{Lap}(\frac{1}{\epsilon})$.

As mentioned earlier, any multi-dimensional range query can be thought of as a rectangle in multi-dimensional space. Furthermore, it is not hard to see that any rectangle can be broken down into $O(\log T)^d$ basic rectangles. For example, Figure 2 illustrates a two-dimensional case. A 2-dimensional range query is made, i.e., $B = [2, 5] \times [3, 5]$. As shown in the figure, the rectangle B can be broken down into $2 \times 3 = 6$ basic rectangles. Therefore, the answer to the range query can be obtained by summing up the noisy count corresponding to each of these basic rectangles.

We now perform the privacy and utility analysis. Recall that in any dimension i, any single point $x \in [T]$ is involved in $O(\log T)$ intervals in $\mathcal{I}_i$. Moreover, any interval in $[1, T]$ can be expressed as the disjoint union of $O(\log T)$ intervals in $\mathcal{I}_i$. Therefore, we can conclude that changing any single entry in the database σ can affect at most $O(\log T)^d$ noisy p-sums, and any rectangle $B \in \mathcal{B}$ can be expressed as the disjoint union of $O(\log T)^d$ rectangles in $\mathcal{I}$, each of which contributes an independent copy of $\mathsf{Lap}(\frac{1}{\epsilon})$ to the error for query B.

Using the p-sum framework, it follows that the resulting mechanism is $O(\log T)^d \cdot \epsilon$-differentially private, and moreover for every $B \in \mathcal{B}$, is $(O(\frac{1}{\epsilon}) \cdot (O(\log T))^{0.5d} \cdot \log \frac{1}{\delta}, \delta)$-useful for query B. After rescaling ϵ, we have an ϵ-differentially private range query mechanism.

THEOREM 7.1. *Let $\epsilon > 0$. For static databases of the form $\sigma : [T]^d \rightarrow \{0, 1\}$, there exists a range query mechanism $\mathcal{M}$ that is ϵ-differentially private, and for each rectangle $B \in \mathcal{B}$, and $0 < \delta < 1$, the mechanism $\mathcal{M}$ is $(O(\frac{1}{\epsilon}) \cdot (O(\log T))^{1.5d} \cdot \log \frac{1}{\delta}, \delta)$-useful for query B.*

8. CONCLUSION AND OPEN PROBLEMS

We consider how a website or data aggregator can privately and continually release new statistics when the database is evolving over time. We propose an ϵ-differentially private continual counter which has only poly-log error with respect to the time.

This represents an exciting and important new setting for differential privacy, as numerous websites adopt the practice of releasing new user statistics over time. A promising direction for future work is the design of a broad class of private streaming algorithms which continually output data. In particular, as counting is a basic building block in many streaming and statistical algorithms, it would be interesting to explore how our basic counting primitive aid the design of other streaming and statistical analysis algorithms.

Another open problem is to explore whether there exist any functions are particularly difficult to release privately in the continual setting, but easy to do so in the offline setting.

It would also be interesting to implement the theoretic results and study their feasibility in real-world applications.

ACKNOWLEDGMENTS

We would like to thank John Bethencourt, Arvind Narayanan, Aaron Roth and Emil Stefanov for helpful discussions and insightful feedback. We gratefully acknowledge the anonymous reviewers for their valuable comments and suggestions.

REFERENCES

CALANDRINO, J. A., NARAYANAN, A., FELTEN, E. W., AND SHMATIKOV, V. 2009. Don't review that book: Privacy risks of collaborative filtering. Manuscript.

DEMAINE, E. D., LÓPEZ-ORTIZ, A., AND MUNRO, J. I. 2002. Frequency estimation of internet packet streams with limited space. In *ESA '02: Proceedings of the 10th Annual European Symposium on Algorithms*.

DINUR, I. AND NISSIM, K. 2003. Revealing information while preserving privacy. In *PODS*.

DWORK, C. 2006. Differential privacy. Invited talk at *ICALP*.

DWORK, C. 2008. Differential privacy: A survey of results. In *TAMC*.

DWORK, C. 2009. The differential privacy frontier. In *TCC*.

DWORK, C. 2010a. Differential privacy in new settings. Invited presentation at *ACM-SIAM Symposium on Discrete Algorithms (SODA)*.

DWORK, C. 2010b. A firm foundation for private data analysis. In *Communications of the ACM*.

DWORK, C., MCSHERRY, F., NISSIM, K., AND SMITH, A. 2006. Calibrating noise to sensitivity in private data analysis. In *TCC*.

DWORK, C., NAOR, M., PITASSI, T., AND ROTHBLUM, G. N. 2010. Differential privacy under continual observation. In *STOC*.

DWORK, C., NAOR, M., PITASSI, T., ROTHBLUM, G. N., AND YEKHANIN, S. 2010. Pan-private streaming algorithms. In *Innovations in Computer Science (ISC)*.

DWORK, C. AND YEKHANIN, S. 2008. New efficient attacks on statistical disclosure control mechanisms. In *CRYPTO*.

HAY, M., RASTOGI, V., MIKLAU, G., AND SUCIU, D. 2010. Boosting the accuracy of differentially private histograms through consistency. *PVLDB 3,* 1, 1021–1032.

JONES, R., KUMAR, R., PANG, B., AND TOMKINS, A. 2008. Vanity fair: privacy in querylog bundles. In *CIKM*.

KOROLOVA, A., KENTHAPADI, K., MISHRA, N., AND NTOULAS, A. 2009. Releasing search queries and clicks privately. In *WWW*.

LI, C., HAY, M., RASTOGI, V., MIKLAU, G., AND MCGREGOR, A. 2010. Optimizing linear counting queries under differential privacy. In *PODS*. 123–134.

MANKU, G. S. AND MOTWANI, R. 2002. Approximate frequency counts over data streams. In *VLDB*.

MCSHERRY, F. AND MIRONOV, I. 2009. Differentially private recommender systems: building privacy into the netflix prize contenders. In *KDD*.

METWALLY, A., AGRAWAL, D., AND ABBADI, A. E. 2005. Efficient computation of frequent and top-k elements in data streams. In *ICDT*.

NARAYANAN, A. AND SHMATIKOV, V. 2008. Robust de-anonymization of large sparse datasets. In *IEEE Symposium on Security and Privacy*.

WARNER, S. L. 1965. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*.

XIAO, X., WANG, G., AND GEHRKE, J. 2010. Differential privacy via wavelet transforms. In *ICDE*. 225–236.

YEGANOVA, L. AND WILBUR, W. 2009. Isotonic regression under lipschitz constraint. *Journal of Optimization Theory and Applications 141*, 429–443. 10.1007/s10957-008-9477-0.