LDP-IDS: Local Differential Privacy for Infinite Data Streams

Xuebin Ren
Xi'an Jiaotong Univeristy
Xi'an, Shaanxi 710049, China
xuebinren@mail.xjtu.edu.cn

Liang Shi
Xi'an Jiaotong University
Xi'an, Shaanxi 710049, China
sl1624@stu.xjtu.edu.cn

Weiren Yu
University of Warwick
Coventry CV4 7AL, United Kingdom
weiren.yu@warwick.ac.uk

Shusen Yang*
Xi'an Jiaotong University
Xi'an, Shaanxi 710049, China
shusenyang@mail.xjtu.edu.cn

Cong Zhao
Imperial College London
London SW7 2AZ, United Kingdom
c.zhao@imperial.ac.uk

Zongben Xu
Xi'an Jiaotong University
Xi'an, Shaanxi 710049, China
zbxu@mail.xjtu.edu.cn

ABSTRACT

Local differential privacy (LDP) is promising for private streaming data collection and analysis. However, existing few LDP studies over streams either apply to finite streams only or may suffer from insufficient protection. This paper investigates this problem by proposing LDP-IDS, a novel w-event LDP paradigm to provide practical privacy guarantee for infinite streams. By constructing a unified error analysis, we adapt the existing budget division framework in centralized differential privacy (CDP) for LDP-IDS, which however incurs prohibitive noise and expensive communication cost. To this end, we propose a novel and extensible framework of population division and recycling, as well as online adaptive population division algorithms for LDP-IDS. We provide theoretical guarantees and demonstrate, through extensive discussions, that our proposed framework not only achieves significant reduction in utility loss and communication overhead, but also enjoys great compatibility for varied analytic tasks and flexibility of incorporating ideas of many existing stream algorithms. Extensive experiments on synthetic and real-world datasets validate the high effectiveness, efficiency, and flexibility of our proposed framework and methods.

CCS CONCEPTS

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

KEYWORDS

Data Streams; Differential Privacy; Local Differential Privacy; Budget Division; Population Division

ACM Reference Format:
Xuebin Ren, Liang Shi, Weiren Yu, Shusen Yang, Cong Zhao, and Zongben Xu. 2022. LDP-IDS: Local Differential Privacy for Infinite Data Streams. In *Proceedings of the 2022 International Conference on Management of Data (SIGMOD '22), June 12–17, 2022, Philadelphia, PA, USA.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3514221.3526190

*The corresponding author.

Table 1: Summary of DP research on data streams

	Event-level	**User-level**	**w-event level**
CDP	[5–7, 17, 18]	[22–24]	[29, 43, 48, 49]
LDP	[26, 45]	[4, 20]	Our work

1 INTRODUCTION

Differential privacy (DP) has emerged as the de-facto standard for privacy protection with rigorous guarantee. DP for streams has also attracted extensive interests. According to the granularity, these studies can be broadly classified into three categorizes: *event-level*, *user-level* and *w-event privacy*. Early researches mainly focus on *event-level privacy* for *finite streams* [5–7, 17, 18] and *user-level privacy* for *infinite streams* [22–24]. However, the former that hides any single event in streams may be insufficient for users' coarse-grained privacy, while the latter that protects any user's occurrence in infinite streams is impractical for realistic scenarios [29]. To break the dilemma, *w-event privacy* for infinite streams is proposed [29], which aims to guarantee ϵ-DP for any time window consisting of w consecutive time instances (or timestamps for simplicity). Due to meaningful protection and applicability, w-event privacy has become the research trend and achieved fruitful results [39, 43, 48, 49]. Nonetheless, these studies are based on *central/centralized differential privacy* (CDP), which relies on a trusted aggregator (or server) and is prone to *honest-but-curious* adversaries.

Recently, *local differential privacy* (LDP) [13, 14, 28] has demonstrated a great potential in accomplishing analytic tasks [8, 21, 32, 33, 35, 36, 38, 41, 44, 46, 47, 50, 51], without relying on a trusted aggregator. Unlike CDP, LDP has the advantage of guaranteeing massive end users' privacy locally, and thereby has been successfully deployed by many well-known corporations, e.g., Google [21], Microsoft [12], Apple [11], and Uber [25]. Contemporary studies on LDP mainly focus on static (non-streaming) data analysis, including frequency [1, 21, 27, 44] and mean estimation [42, 50]. For evolving (streaming) data analytics, there are only a little work, including event-level LDP for infinite streams [26, 45] and user-level LDP for finite streams [4, 20]. Under event-level privacy, Joseph *et al.* [26] propose THRESH, which aims at reducing privacy loss at time slots with no significant population-wide updates. Despite being compatible to infinite streams, event-level LDP lacks consideration of users' coarse-grained data. For user-level LDP, Bao *et al.* [4] present a correlated Gaussian mechanism CGM via utilizing autocorrelations in streams. However, under the analytic Gaussian mechanism, CGM achieves only approximate LDP (i.e., (ϵ, δ)-LDP) rather than pure ϵ-LDP, and is limited to finite streams only, meaning that the service has to be restarted periodically for infinite stream scenarios. Table 1 summarizes DP studies on data streams from both aspects of privacy granularity and applicable architecture. To the best of our knowledge, there is no prior work on w-event LDP, which can persistently provide strong and practical protection for indefinitely streaming data collection.

In this paper, we propose LDP-IDS, a pure ϵ-LDP based paradigm over *infinite streams* under the framework of *w-event privacy*. There are three technical challenges for LDP-IDS:

- **No access to raw data**. In CDP studies [29], to reduce overall noise, it is common to mainly update at remarkable timestamps or assign different budget at different timestamps according to the sparsity in raw streams. However, this is difficult in LDP protocols, since the aggregator no longer has access to the raw data streams, which have to be perturbed at the user's end locally.
- **Utility loss in budget division**. Even if some methods can be formulated to mine the characteristics of raw streams underlying the LDP perturbed streams, the budget division methodology, commonly used in CDP, is not efficient in LDP. Compared with CDP noise, LDP noise is more sensitive to budget division and perturbation with low LDP budget is much more noisy [44, 47].
- **High communication cost**. As streaming data generates, massive users persistently release their perturbed data to the aggregator at each timestamp. That often causes high communication cost for resource-constrained devices. It is desirable to consider the communication efficiency in designing LDP mechanisms.

To address the above challenges, we propose LDP-IDS, an LDP paradigm and corresponding methods for infinite data streams. Specifically, in this paper, we make the following contributions:

- We first formulate the problem of infinite streaming data collection with LDP, which aims at realizing accurate and efficient statistical analysis over LDP perturbed streams while providing meaningful privacy protection (i.e., w-event LDP) for infinite streams and not relying on a trusted server (§. 3).
- We construct a unified error analysis for streaming data analytics with LDP. Based on that, we develop the budget division framework with theoretical guarantees for achieving LDP over infinite streams. We present two online adaptive budge division methods, which allocate budget according to non-deterministic stream sparsity. Compared to naive methods, the online adaptive methods can effectively improve data utility (§. 4).
- We propose a novel framework of population division and recycling for LDP-IDS, which achieves significantly higher utility and less communication overheads (§. 5). To the best of our knowledge, this is the first study on population division-based LDP for infinite streams. By building an analogy between budget and population division, we design online adaptive population division methods with further performance improvement. Besides, we discuss the generality and extend the framework to other state-of-the-art stream algorithms (§. 6).

We implemented all proposed LDP algorithms for streams and conducted extensive experiment evaluation on both synthetic and real-world datasets (§. 7). Our experimental results validate that, compared with the budget division algorithms, population division and recycling-based algorithms achieve significant reduction in utility loss, event detection error, and communication overhead. Furthermore, the proposed framework of online adaptive population division for LDP-IDS has demonstrated great compatibility for varied analytic tasks and high flexibility of incorporating with existing stream algorithms.

2 PRELIMINARIES

In this section, we first introduce w-event privacy and its CDP studies. Then, we introduce LDP and frequency oracles.

2.1 w-event Privacy in Centralized Setting

w-event privacy can strike a nice balance between event-level privacy for infinite streams and user-level privacy for finite streams.

We first give the notion about *stream prefix* and *neighboring streams*. A stream prefix of an infinite series $S = (D_1, D_2, \ldots)$ at timestamp t is defined as $S_t = (D_1, D_2, \ldots, D_t)$, where D_i is a snapshot of the stream at i. Let w be a positive integer, two stream prefixes S_t, S'_t are called *w-neighboring*, if for each $S_t[i], S'_t[i]$ such that $i \in [t]$ and $S_t[i] \neq S'_t[i]$, it holds that $S_t[i], S'_t[i]$ are neighboring; and for each $S_t[i_1], S_t[i_2], S'_t[i_1], S'_t[i_2]$ with $i_1 \leq i_2$, $S_t[i_1] \neq S'_t[i_1]$ and $S_t[i_2] \neq S'_t[i_2]$, it holds that $i_2 - i_1 + 1 \leq w$.

Definition 2.1 (w-event Privacy [29]). Let $\mathcal{M}$ be a mechanism that takes as input a stream prefix of arbitrary size and $\mathcal{O}$ denote the set of all possible outputs of $\mathcal{M}$. $\mathcal{M}$ satisfies w-event ϵ-DP (or, simply, w-event privacy) if for all sets $O \subseteq \mathcal{O}$, all w-neighboring stream prefixes S_t, S'_t, and all t, $\Pr[\mathcal{M}(S_t) \in O] \leq e^{\epsilon} \cdot \Pr[\mathcal{M}(S'_t) \in O]$.

w-event privacy guarantees ϵ-DP in any sliding window of size w. Or, for any w-event private mechanism, ϵ can be viewed as the total available privacy budget in any sliding window of size w [29].

2.2 Existing Methods with w-event CDP

A naive method is to evenly apply ϵ/w-DP histogram release mechanism at every timestamp. Unfortunately, with the increase of w, the allocated budget becomes much small, which causes large perturbation noise at each timestamp. Another simple method is to release an ϵ-DP fresh histogram at one timestamp while other timestamps in a window is directly approximated with this result. However, the fixed sampling strategy cannot accurately follow the update patterns in the dynamic stream, thus leading to large errors.

BD (budget distribution) and BA (budget absorption) are benchmark adaptive methods for infinite stream release with w-event CDP[29]. Both BD and BA can be summarized into three components: *private dissimilarity calculation*, *private strategy determination*, and *privacy budget allocation*. In *private dissimilarity calculation*, a dissimilarity *dis* between the current $\mathbf{c}_t$ and the last update $\mathbf{c}_l$ is computed and perturbed with some fixed *dissimilarity budget* $\epsilon_{t,1}$. In *private strategy determination*, some *publication budget* $\epsilon_{t,2}$ is assigned (How to assign is designed in *privacy budget allocation*) for potential publication of noisy statistic, which can derive a potential publication error *err*. Then, *dis* and *err* are compared to decide the private strategy for current release. If $err < dis$, publish with perturbation (i.e., $\mathbf{r}_t = \mathbf{c}_t + \langle Lap(1/\epsilon_{t,2})\rangle^d$); otherwise, approximate by the previous release (i.e., $\mathbf{r}_t = \mathbf{c}_l$). In above process, $\epsilon_{t,1}$ is fixed for each timestamp, but $\epsilon_{t,2}$ is assigned based on different rules in

BD and BA. In BD, $\epsilon_{t,2}$ is distributed in an exponentially decaying way to the timestamps where a publication is chosen, and reuses the budget spent in timestamps out of the current sliding window. While in BA, $\epsilon_{t,2}$ is uniformly assigned first and then unused budget is absorbed at timestamps where approximation is chosen.

2.3 Local Differential Privacy (LDP)

In the LDP paradigm, $\mathcal{M}$ is a randomized mechanism that perturbs each user's input v and sends it to the central aggregator, who estimates the true aggregate statistics from the perturbed data.

Definition 2.2 (Local Differential Privacy). A mechanism $\mathcal{M}$ satisfies ϵ-local differential privacy (i.e., ϵ-LDP), if and only if, for any input v and v' in domain $Dom(\mathcal{M})$ and any O in the set of all possible outputs of $\mathcal{M}$, we have $\Pr[\mathcal{M}(v) \in O] \le e^{\epsilon} \cdot \Pr[\mathcal{M}(v') \in O]$.

LDP ensures the aggregator cannot infer the input with high confidence. As a variant, it inherits the DP properties, including sequential/parallel composition and post-processing theorems [19][31].

2.4 LDP Protocols for Frequency Estimation

LDP data analyses are commonly built on some frequency oracle (FO) protocols, which enable frequency estimation of any value v in a given domain $\Omega = \{\omega_1, \omega_2, \ldots, \omega_d\}$ of size $d = |\Omega|$ with ϵ-LDP.

GRR. A common LDP FO protocol is *Generalized Randomized Response (GRR)*. The idea of GRR method is that with a private data $v \in \Omega$, each user sends the true value to the central aggregator with probability $p = \frac{e^{\epsilon}}{e^{\epsilon}+d-1}$, and randomly sends other value in the candidate set $\Omega \setminus \{v\}$ with probability $q = \frac{1}{e^{\epsilon}+d-1}$. The aggregator estimates the *frequency* of each distinct item v or $\omega_k \in \Omega$, denoted as $\mathbf{c}[k]$, as follows. It first counts the frequency of ω_k in perturbed data, which is denoted as $\mathbf{c}'[k]$. Then, assuming n is the number of participant users, the estimated frequency $\overline{\mathbf{c}}[k]$ of ω_k through GRR protocol can be obtained as $\overline{\mathbf{c}}_{\text{GRR}}[k] = (\mathbf{c}'[k]/n - q)/(p - q)$. It is shown in [44] that this is an unbiased estimation of the true frequency, with the variance

$$\text{Var}[\overline{\mathbf{c}}_{\text{GRR}}[k]; \epsilon, n] = \frac{d - 2 + e^{\epsilon}}{n \cdot (e^{\epsilon} - 1)^2} + \frac{\mathbf{c}[k] \cdot (d-2)}{n \cdot (e^{\epsilon} - 1)} \tag{1}$$

where $\mathbf{c}[k]$ is the frequency of ω_k and there is $\sum_{k=1}^{d} \mathbf{c}[k] = 1$.

OUE. There is another efficient LDP protocol *Optimal Unary Encoding* (OUE), in which an unary encoding is applied before randomization [51]. This protocol first encodes an input value $v \in \Omega$ into a binary string of length d with the v-th bit set as 1 and all other bits as 0. Then OUE applies GRR to each bit, but perturbs 1's and 0's differently, i.e., the bit 1 is perturbed as a coin toss while bit 0 is perturbed with the maximum allowed budget ϵ. To estimate the frequency of $v \in \Omega$, the aggregator collects all perturbed bit strings and counts the number of reports (denoted as $\mathbf{c}'[k]$) with the bit corresponding to v set to 1. Then, its unbiased estimation can be derived as $\overline{\mathbf{c}}_{\text{OUE}}[k] = 2(\mathbf{c}'[k] - n \cdot q)/(1 - 2q)$ with the variance

$$\text{Var}[\overline{\mathbf{c}}_{\text{OUE}}(k); \epsilon, n] = \frac{4e^{\epsilon}}{n \cdot (e^{\epsilon} - 1)^2} + \frac{\mathbf{c}[k]}{n} \tag{2}$$

Ada. To minimize estimation error, it is proposed to adaptively choose the above two protocols according to the domain size and privacy budget. If $d < 3e^{\epsilon} + 2$, GRR is used; otherwise, OUE is

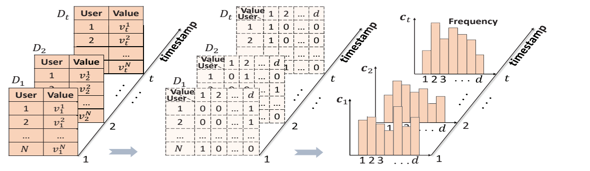


Figure 1: Histogram release in a streaming setting

adopted. This is referred to as Adaptive FO (Ada). We also adopt it in our paper for better utility.

Note that, given the domain size d and ϵ, the variance of FOs is determined and can all be expressed as a function of parameter ϵ and population n, as comprehensively analyzed in [10]. For conciseness, without specifying the FO used, we use $V(\epsilon, n)$ to represent the estimation variance from n users with budget ϵ. With a different FO, the communication cost for a single report is different. For example, the number of communication bits for each report is $O(\log d)$ for GRR and $O(d)$ for OUE, i.e., at most $O(d)$ for Ada.

3 MODELS AND PROBLEM DEFINITION

In this section, we propose LDP-IDS, a novel LDP paradigm for infinite streams under the framework of w-event privacy.

Data Model. We consider a system consisting of a central server and n distributed users $U = \{u_1, \ldots, u_n\}$. At discrete timestamps, the server collects data over some attributes (e.g., setting preference or battery usage of devices) from these users, and conducts statistical analysis on the collected data continuously. For a certain attribute, let v_t^j represent the report of user u_j at timestamp t and v_t^j come from a domain Ω, which can be either categorical (e.g, $\Omega = \{\omega_1, \omega_2, \ldots, \omega_d\}$ with the cardinality of d) or numerical (e.g., $\Omega = [0, B]$ with a maximal differential bound B). Then, as time evolves, each user inherently has an infinite data stream $V^j = (v_1^j, v_2^j, \ldots)$, which is sensitive and prohibited to directly share with the server. Meanwhile, at every timestamp t, the server aims to obtain and release the estimate $\mathbf{r}_t$ of the true aggregate statistics $\mathbf{c}_t$ (e.g. histogram shown in Fig. 1) over all n users' data continuously.

Privacy Model. In the LDP setting, instead of reporting v_t^j, each user would send a perturbed representation $\overline{v}_t^j$ (ϵ-LDP ($\epsilon > 0$) or nothing (equivalent to 0-LDP) at each timestamp t. An Ada-based local sanitation process at the user-side is shown in Algorithm 1. Considering the stream infinity, similar to w-event privacy in the centralized setting, users also wish to adopt a meaningful privacy paradigm in the LDP setting. We naturally extend the definition of w-event privacy to the local setting. Before that, we first define the notion of w-neighboring in the local setting.

Algorithm 1: LocSan (at **User-side**):

```
Input: timestamp $t$, budget $\epsilon$, domain $\Omega$ of size $d$
Output: Perturbed report $\overline{v}_t^j$
1 Obtain the current raw report $v_t^j$;
2 if $d < 3e^{\epsilon} + 2$ then
3 |   $\mathcal{M} \leftarrow$ GRR;
4 else
5 |   $\mathcal{M} \leftarrow$ OUE;
6 end
7 $\overline{v}_t^j \leftarrow \mathcal{M}(v_t^j, \epsilon, \Omega)$;
8 Submit the perturbed report $\overline{v}_t^j$
```

Definition 3.1 (w-neighboring). Let V_t and V'_t denote two stream prefixes over a domain Ω^t. Let w be a positive integer. V_t and V'_t are w-neighboring, if for each $V_t[i_1], V_t[i_2], V'_t[i_1], V'_t[i_2]$ with $i_1 \le i_2$, $V_t[i_1] \ne V'_t[i_1]$ and $V_t[i_2] \ne V'_t[i_2]$, it holds that $i_2 - i_1 + 1 \le w$.

That is to say, if two stream prefixes are w-neighboring, then their elements are *the same* while all their *same* elements consist of a window of up to w timestamps. This is slightly different from the definition in the central setting.

Definition 3.2 (w-event LDP). Let $\mathcal{M}$ be a mechanism that takes as input stream prefix $V_t = (v_1, v_2, \dots, v_t)$ consisting of a single user's arbitrary number of consecutive input value v_t. Also let $\mathcal{O}$ be the set of all possible outputs of $\mathcal{M}$. $\mathcal{M}$ satisfies w-event ϵ-LDP (i.e., w-event LDP) if for any w-neighboring stream prefixes V_t, V'_t, any $O \subseteq \mathcal{O}$ and all t, it holds that $\Pr[\mathcal{M}(V_t) \in O] \le e^{\epsilon}\Pr[\mathcal{M}(V'_t) \in O]$.

Problem Definition. We consider the following two typical classes of aggregate statistics over infinite streams.

- *Frequency estimation over categorical domain* aims to estimate the frequency histogram $\mathbf{c}_t = \langle \mathbf{c}_t[1], \mathbf{c}_t[2], \dots, \mathbf{c}_t[d] \rangle$ of all d possible values over $\Omega = \{\omega_1, \omega_2, \dots, \omega_d\}$, where $\mathbf{c}_t[k] = \frac{1}{n}\sum_j \mathbb{1}_{\{k|v_t^j=\omega_k\}}(k)$ is the frequency of $\omega_k \in \Omega$ and $\mathbb{1}_X(k)$ is an indicator function that equals to 1 if $k \in X$, and 0 otherwise.
- *Mean estimation over numerical domain* aims to estimate the mean value $c_t = \frac{1}{n}v_t^j$ over all n users' data $v_t^j \in [0, B]$. Without loss of generality, we also use $\mathbf{c}_t = \{c_t\}$ with cardinality $d = 1$ to denote the mean-value statistics.

Since different LDP aggregate statistics are orthogonal and work in a similar way in the stream setting, we mainly focus on frequency estimation in the remainder of this paper, and then discuss the generality and extension to mean estimation in Sec. 6.1.

Therefore, our goal is to design an w-event ϵ-LDP solution that helps the server to collect data and release an estimated histogram $\mathbf{r}_t = \langle \mathbf{r}_t[1], \mathbf{r}_t[2], \dots, \mathbf{r}_t[d] \rangle$ at each timestamp t where $\mathbf{r}_t[k]$ denotes the estimated frequency or mean in the domain. Particularly, we aim to minimize the average distance between the estimate sequence $\mathbf{R}_t = (\mathbf{r}_1, \dots, \mathbf{r}_t)$ and the true one $\mathbf{C}_t = (\mathbf{c}_1, \dots, \mathbf{c}_t)$ while satisfying w-event LDP over infinite data streams.

4 BUDGET DIVISION METHODOLOGY

In this section, we present a budget division-based framework and several baselines for streaming data collection with LDP.

4.1 Budget Division for LDP

Inspired by the studies in the centralized setting, the following theorem can be derived to enable a w-event LDP mechanism to view ϵ as the total privacy budget in any sliding window of size w, and allocate portions of it across timestamps, as shown in Fig. 2(a).

Theorem 4.1. *Let $\mathcal{M}$ be a mechanism that takes as input stream prefix V_t consisting of a single user's arbitrary number of consecutive input value v_i, i.e., $V_t[i] = v_i$, and outputs a transcript $o = (o_1, o_2, ..., o_t) \in Range(\mathcal{M})$. Suppose that we can decompose $\mathcal{M}$ into t mechanisms $\mathcal{M}_1, \mathcal{M}_2, ..., \mathcal{M}_t$, such that $\mathcal{M}_i(v_i) = o_i$, each $\mathcal{M}_i$ generates independent randomness and achieves ϵ_i-LDP. Then, $\mathcal{M}$ satisfies w-event ϵ-LDP if for any user and any timestamp $i \in [t]$, there is $(\sum_{\tau=i-w+1}^{i} \epsilon_\tau) \le \epsilon$.*

Proof. See Appendix B.1 in [37]. □

By Theorem 4.1, some straightforward LDP budget division approaches can be summarized to solve the problem defined in Sec. 3.

LDP Budget Uniform Method (LBU). One straightforward approach is to uniformly assign LDP budget ϵ to all w timestamps in the sliding windows. At each timestamp, each user reports with an LDP protocol using the fixed budget ϵ/w for satisfying w-event LDP. Since $\mathbf{r}_t$ is an unbiased estimate of $\mathbf{c}_t$, the mean square error (MSE) between the true stream prefix $C_t = (\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_t)$ and the released one $R_t = (\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_t)$, equals to the estimate variance of $\mathbf{r}_t$, i.e., $\text{MSE}_{\text{LBU}} = \text{Var}[\mathbf{r}_t; \epsilon/w, n] = V(\epsilon/w, n)$. If w is large, the budget allocated at each timestamp is small, leading to a large noise. Communication cost, measured by the number of communication bits between the users and the server per timestamp, is $O(n \log d)$ for GRR and $O(nd)$ for OUE, i.e., at most $O(nd)$ for Ada.

LDP Sampling Method (LSP). Each user invests the entire budget ϵ on a single (sampling) timestamp within the window, while saving budget for the next $w - 1$ timestamps via approximation. At the last sampling timestamp l, the MSE of LSP equals to estimation variance $V(\epsilon, n)$. For non-sampling timestamps, it equals to the sum of the variance of last release at the sampling timestamp (i.e., $\text{Var}[\mathbf{r}_l]$ [1]), and the variance of true statisics at the current timestamp t from that at the last sampling timestamp l (i.e., $(\mathbf{c}_t - \mathbf{c}_l)^2$, which is data dependent). Therefore, the MSE of LSP in a window of size w can be calculated as $\text{MSE}_{\text{LSP}} = V(\epsilon, n) + \frac{1}{w}\sum_{k=1}^{w-1}(\mathbf{c}_t - \mathbf{c}_{t+k})^2$. An implicit assumption motives this method is that $\mathbf{c}_t$ (or D_t) in the stream does not fluctuate too much. Therefore, for streams with few changes, LSP may work better by saving up privacy budgets; otherwise, the estimation error on those skipped timestamps may become excessively large. Besides budget saving, LSP reduces the total communication bits per timestamp to $O(nd/w)$.

Considering the non-deterministic sparsity in data streams, both LBU and LSP can not achieve better utility in general cases.

4.2 Adaptive Budget Division Methods

In this subsection, we propose two adaptive methods by constructing a unified distortion analysis under LDP.

BD/BA [29] summarized in Sec. 2.2 inspire us that higher utility can be achieved by adaptively allocating privacy budget in data streams. However, in the local setting, since the central server cannot observe individuals' reports or directly obtain the true $\mathbf{c}_t$, the design of such LDP solutions is challenging. In particular, it is infeasible to accomplish the private dissimilarity calculation or data publication by adding noise over the true statistics, but we need to use LDP protocols to do so. However, using an LDP protocol, it remains unclear how to model the *dissimilarity dis* and *publication error err* under LDP for empirically optimal strategy determination.

4.2.1 ***Private dissimilarity estimation***. To address the above challenges, we first redefine the dissimilarity measure dis^* as the square error between the true statistics $\mathbf{c}_t$ of current timestamp and the previous release $\mathbf{r}_l$, i.e., $dis^* = \frac{1}{d}\sum_{k=1}^{d}(\mathbf{c}_t[k] - \mathbf{r}_l[k])^2$. Then, in $\mathcal{M}_{t,1}$, we aim to obtain the dissimilarity dis^* privately, i.e., from users' LDP perturbed data using the dissimilarity budget $\epsilon_{t,1}$.

[1]For simplicity, we use $\text{Var}[\mathbf{r}_l]$ to denote the average variance over d dimensions of vector$\mathbf{r}_l$, or $\text{Var}[\mathbf{r}_l] = \frac{1}{d}\sum_{k=1}^{d}\text{Var}(\mathbf{r}_l[k])$.

Theorem 4.2. *Let $\bar{\mathbf{c}}_{t,1}$ denote the unbiased estimate of $\mathbf{c}_t$ from an ϵ-LDP mechanism over the perturbed data in $\mathcal{M}_{t,1}$. Then, the following dissimilarity measure*

$$dis = \frac{1}{d}\sum\nolimits_{k=1}^{d}(\bar{\mathbf{c}}_{t,1}[k] - \mathbf{r}_l[k])^2 - \frac{1}{d}\sum\nolimits_{k=1}^{d} Var(\bar{\mathbf{c}}_{t,1}[k]). \quad (3)$$

is ϵ-LDP and an unbiased estimation of dis^* above defined.

Proof. See Appendix B.2 in [37]. □

Therefore, the dissimilarity can be calculated from $\bar{\mathbf{c}}_{t,1}$ while satisfying $\epsilon_{t,1}$-LDP. In the left term $\frac{1}{d}\sum_{k=1}^{d}(\bar{\mathbf{c}}_{t,1}[k] - \mathbf{r}_l[k])^2$ of Eq. (3), $\bar{\mathbf{c}}_{t,1}$ is obtained from an LDP protocol while $\mathbf{r}_l$ is publicly known. The right term $\frac{1}{d}\sum_{k=1}^{d}\text{Var}(\bar{\mathbf{c}}_{t,1}[k])$ denoted as $V(\epsilon_{t,1}, n)$, can be estimated from the population n and LDP budget $\epsilon_{t,1}$.

4.2.2 ***Private strategy determination***. The key to choose the strategy of approximation or publication is to compare the dissimilarity (i.e., the potential *approximation error*) with the potential *publication error*. Considering that LDP protocols and their analysis are different from those in the CDP setting, the LDP-based publication error should also be re-formulated.

Here, in the LDP setting, considering that *dis* defined above is an L_2 distance measure, we propose to use Mean Square Error (MSE) to measure the potential publication error, denoted as *err*. Suppose $\bar{\mathbf{c}}_{t,2}$ is the histogram estimated via LDP protocol (e.g., GRR), the estimation error can be measured as $err = \frac{1}{d}\sum_{k=1}^{d}(\bar{\mathbf{c}}_{t,2}[k]-\mathbf{c}_t[k])^2$. Since $\bar{\mathbf{c}}_{t,2}$ is an unbiased estimation of $\mathbf{c}_t$, i.e., $\mathbb{E}(\bar{\mathbf{c}}_{t,2}) = \mathbf{c}_t$, there is

$$err = \frac{1}{d}\sum\nolimits_{k=1}^{d}\text{Var}(\bar{\mathbf{c}}_{t,2}[k]) \quad (4)$$

which denotes as $V(\epsilon_{t,2}, n)$ and can be calculated from the population n and *publication budget* $\epsilon_{t,2}$. For example, taking GRR as an example of the used LDP protocol, it can be written as $err = \frac{1}{d}\sum_{k=1}^{d}\text{Var}(\bar{\mathbf{c}}_{t,2}[k]) = \frac{d-2+e^{\epsilon_{t,2}}}{n(e^{\epsilon_{t,2}}-1)^2} + \frac{d-2}{n(e^{\epsilon_{t,2}}-1)}$. Note that, *err* is independent of the true frequency value $\mathbf{c}[k]$ in Eq. (1).

Based on above formulations, an empirically optimal strategy at current timestamp t can be determined as follows.

- If $dis < err$, the approximation strategy is chosen. For example, the server can directly publish the last released value without actual consumption of the publication budget $\epsilon_{t,2}$.
- Otherwise, the publication strategy is chosen. Each user reports value via an LDP protocol using the publication budget $\epsilon_{t,2}$ to the server, who releases a fresh estimate $\bar{\mathbf{c}}$.

4.2.3 ***Privacy budget allocation***. From the high level, we evenly divide the entire budget in a time window, ϵ, for two components: private dissimilarity estimation and private strategy determination. Therefore, the entire dissimilarity budget and publication budget in a time window satisfies $\sum_{i=t-w+1}^{t}\epsilon_{i,1} \le \epsilon/2$, $\sum_{i=t-w+1}^{t}\epsilon_{i,2} \le \epsilon/2$. In the private dissimilarity estimation, the dissimilarity budget is divided evenly to each timestamp in the time window, i.e., $\epsilon_{i,1} = \epsilon/2w$. However, we aim to invest the publication budget economically to the timestamps, which leads to two different methods, **LDP budget distribution** (LBD) and **LDP budget absorption** (LBA).

In **LBD**, the publication budget is distributed in an exponentially decreasing way to the timestamps where a publication occurs. At each timestamp t, before private strategy determination, the current remaining publication budget is calculated as, $\epsilon_{\text{rm}} = \epsilon/2 - \sum_{i=t-w+1}^{t-1}\epsilon_{i,2}$. Then, half of the remaining publication budget ϵ_{rm} is assigned as the *potential publication budget* $\epsilon_{t,2}$ for the calculation of potential publication error. If publication strategy is chosen, $\epsilon_{t,2}$ is spent, meaning $\epsilon_{t,2}$ will be removed from the remaining publication budget at the next timestamp. If approximation occurs, $\epsilon_{t,2}$ is then not truly used, i.e., $\epsilon_{t,2} = 0$. Finally, the publication budget $\epsilon_{t-w,2}$ spent in timestamp $t-w$ ($t > w$), out of the current sliding window is recycled. The pseudocode of LBD can be referred to Appendix C.1 in [37].

In **LBA**, the publication budget is uniformly allocated at all timestamps and then unused budget is absorbed at timestamps where the publication strategy is chosen. Initially, the same amount of potential publication budget, $\epsilon/2w$, is evenly assigned to all timestamps in a time window. Then, at timestamp t, if the approximation strategy was chosen at previous timestamps, the unused publication budgets at these timestamps are added to $\epsilon_{t,2}$; if publication was chosen previously, it must nullify the same budget from the succeeding timestamps to ensure the total budget within the active sliding window does not exceed ϵ. The pseudocode of LBA can be referred to Appendix C.2 in [37].

4.3 Analysis

4.3.1 Privacy Analysis. Both LBD and LBA satisfy w-event LDP.

Theorem 4.3. *LBD and LBA satisfy w-event LDP for each user.*

Proof. See Appendix B.3 in [37]. □

4.3.2 Utility Analysis. For simplicity, in both LBD and LBA, we assume there are $m < w$ publications occur at the timestamps $p_1, p_2, ..., p_m$ in the window of size w. Besides, no budget is recycled from past timestamps outside the window, and each publication approximates the same number of skipped/nullified publications. Similar to the analysis of LSP, at any timestamp t, if publication occurs, then the MSE of the release $\mathbf{r}_t$ is $\text{MSE}_{\text{pub}} = \text{Var}[\mathbf{r}_t]$; if approximation is chosen, its MSE equals to the sum of the variance of last release at timestamp l (i.e., $\text{Var}[\mathbf{r}_l]$), and the variance of the true statisics at the current timestamp t from that at timestamp l (i.e., $(\mathbf{c}_t - \mathbf{c}_l)^2$), i.e., $\text{MSE}_{\text{apr}} = \text{Var}[\mathbf{r}_l] + (\mathbf{c}_t - \mathbf{c}_l)^2$. Then we express MSE in a whole time window as follows

$$\text{MSE}_{LBD/LBA} = \frac{1}{w}\left[\frac{w}{m}\sum_{i=1}^{m}\text{Var}[\mathbf{r}_{p_i}] + \sum_{i=1}^{m}\sum_{t=p_i}^{p_{i+1}-1}(\mathbf{c}_t - \mathbf{c}_{p_i})^2\right] \quad (5)$$

where the second term in the bracket solely depends on the underlying data and shows the data-dependent characteristics of LBD and LBA. In the following, we analyze the left term in the bracket.

In LBD, since the budget is distributed to the m publications in an exponentially decreasing way, the budget sequence of $\epsilon_{t,2}$ is then $\epsilon/4, \epsilon/8, \ldots, \epsilon/2^{m+1}$. There is

$$\sum_{i=1}^{m}\text{Var}_{\text{LBD}}[\mathbf{r}_{p_i}] = \sum_{i=1}^{m} V(\epsilon/2^{i+1}, N) < m \cdot V(\epsilon/2^{m+1}, N) \quad (6)$$

where $V(\epsilon, n)$ denotes the estimation variance of an FO from n users' LDP data using budget ϵ. As we can see, with the increase of m, the error of LBD would increase dramatically.

In LBA, due to m publications in the assumption, there are $w - m$ approximations. Since each publication approximates the same number of skipped/nullified publications, there are $\frac{w-m}{2\cdot m}$ skipped (

whose budgets are absorbed) and $\frac{w-m}{2\cdot m}$ nullified publications in average. Then, each publication receives those skipped budget $\frac{(\frac{w-m}{2\cdot m}+1)\cdot\epsilon}{2\cdot w} = \frac{(w+m)\cdot\epsilon}{4\cdot w\cdot m}$ and incurs MSE of $V(\frac{w+m}{4\cdot w\cdot m}\cdot\epsilon, n)$.

$$\sum_{i=1}^{m} \mathrm{Var}_{\mathsf{LBA}}[\mathbf{r}_{p_i}] = m\cdot V(\frac{w+m}{4\cdot w\cdot m}\cdot\epsilon, N) \qquad (7)$$

Compared to LBD, LBA's error increases with m more mildly.

4.3.3 Communication Analysis. In both LBD/LBA, each user has to report twice to the server at publication timestamps and only once at approximation timestamps. Meanwhile, the server has to instruct users to publish or approximate at each timestamp. When there are m publications in a window of size w, the number of communication bits from the users to the server is at most $O(nd(1+m/w))$ while that from the server to the users is at most $O(n\cdot\min(m, w-m)/w)$, or simply $O(n)$, per timestamp. Therefore, the total communication bits per timestamp are $O(n\cdot(d+dm/w+1))$.

5 POPULATION DIVISION METHODOLOGY

In this section, we present a novel idea of population division for LDP over data streams. Then, we propose two population division-based methods for better utility and communication efficiency.

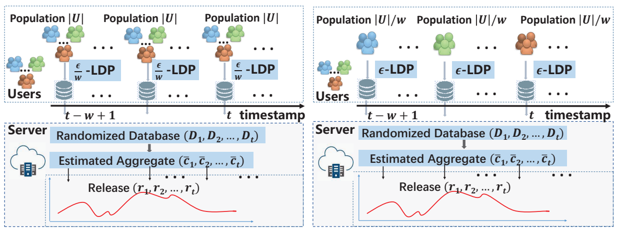
5.1 Population Division & Recycling for LDP

The budget division framework provides a feasible solution to infinite streaming data collection with LDP. However, data utility in LDP scenarios is much more sensitive to budget division than that in CDP. Recall that $V(\epsilon, n)$ denotes the estimation variance of an LDP protocol from n users with privacy budget ϵ. In Eq. (1), with fixed n, $V(\epsilon, n)$ is $O((e^{\epsilon}-1)^{-1})$ in terms of budget ϵ. It increases sharply as the budget assigned to each timestamp becomes small. Recently, several previous studies [44, 47] indicate that it can achieve much smaller overall error by partitioning users into groups and using the entire privacy budget in each group. With the fixed ϵ, $V(\epsilon, n)$ is $O(n^{-1})$ in terms of user population n, which increases much mildly as n becomes small. Hence, it is favorable to divide the population rather than budget in LDP. However, most studies focus on population division in a static setting, it remains unclear how to divide and utilize the given population in an online mode over infinite stream for achieving better utility.

w-event LDP requires that each user can report with ϵ-LDP only once in a time window of size w. Nonetheless, after w timestamps, he/she can report again with the entire ϵ LDP budget. In such a way, we can continue sampling and recycling the users in data collection to achieve population division over infinite streams. Therefore, we adopt this idea in streaming data collection with LDP. Intuitively, a straightforward method can be derived to achieve w-event LDP.

LDP Population Uniform Method (LPU). At the beginning, the central server uniformly assign the users into w disjoint groups, each with roughly n/w users[2]. At each timestamp, it requests a group of users that have never been requested before to report their value. In a window with w timestamps, each group of users will only report once with the entire budget ϵ. And after $w-1$ timestamps, each group users will be requested and report again for the new sliding window. In this case, any user does not report

[2]Precisely, if $n \mod w \neq 0$, it may be $\lfloor n/w \rfloor$ for some groups or $\lfloor n/w \rfloor + 1$ for the rests. For simplicity, we assume n/w for each.



(a) Budget Division-based LDP (b) Population Division-based LDP

Figure 2: Budget division vs. Population division

in each sliding window more than once, thus spending no more than ϵ-LDP budget. Hence, w-event LDP is guaranteed for each user. Fig. 2(b) illustrates the uniform population division method.

THEOREM 5.1. *Given the same FO protocol GRR or OUE, the MSE of LPU is smaller than that of LBU, i.e., $MSE_{LPU} < MSE_{LBU}$.*

PROOF. See Appendix B.4 in [37]. □

Note that, since only a portion of users participate in reporting, the population division methodology can also greatly reduce the communication cost. In LPU, the number of users upload perturbed data at each timestamp is only $1/w$ of the whole population in average. Therefore, the communication cost is $1/w$ of that in LBU.

LDP Population Sampling Method (LSP). LSP in Sec. 4.1 can be also seen as a population division method. Particularly, all users are regarded as to be divided into w groups, in which, one group has the whole population and entire budget while the rests have no user reporting or budget allocation.

However, both LPU and LSP cannot be adaptive to streams with unknown fluctuations, which still limits their utility.

5.2 Adaptive Population Division Methods

LBD/LBA provides a reference framework that improves the utility of baseline methods via adaptively assigning privacy budget according to the non-deterministic sparsity in data streams. In the following, we present two online adaptive population division methods LPD/LPA, which migrates this idea to the population division & recycling framework to further enhance the data utility and reduce the communication cost.

5.2.1 ***Overview.*** For better analogy to LBD/LBA, we still introduce the population division based methods with two sub mechanisms $\mathcal{M}_1$ and $\mathcal{M}_2$. We first evenly partition the whole population U of size n into *dissimilarity user* set U_1 of size $|U_1|$ for $\mathcal{M}_1$ and *publication user* set U_2 of size $|U_2|$ for $\mathcal{M}_2$, each with $\lfloor n/2 \rfloor$ users. Similarly, $\mathcal{M}_1$ performs *private dissimilarity calculation.* Differently, based on population division framework, $\mathcal{M}_2$ accomplishes *private strategies determination* and *user allocation & recycling.*

Private dissimilarity calculation: Sec. 4.2.1 defines the dissimilarity measure *dis* in the LDP setting. In $\mathcal{M}_1$, at each timestamp t, we still aim to estimate the dissimilarity $dis^* = \frac{1}{d}\sum_{k=1}^{d}(\mathbf{c}_t[k] - \mathbf{r}_l[k])^2$ based on Eq. (3). Similar to Sec. 4.2.1, we have to first obtain an unbiased estimation $\bar{\mathbf{c}}_{t,1}$ from the LDP reports. However, based on the population division methodology, it can only be obtained from the LDP reports (using privacy budget ϵ) of the dissimilarity users in $U_{t,1}$ at timestamp t. Particularly, we partition the $\lfloor n/2 \rfloor$

Algorithm 2: LPD (at **Server-side**)

```
   Input: Population $U$ of size $|U| = n$, budget $\epsilon$, window size $w$, domain $\Omega$
   Output: Released statistics $R_t = (\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_t, \ldots)$
1  Initialize available userset $U_A = U$, and $\mathbf{r}_0 = \langle 0, \ldots, 0 \rangle$;
2  for each timestamp t do
       // Sub Mechanism $\mathcal{M}_{t,1}$:
3      Sample users $U_{t,1}$ from $U_A$ with the size of $\lfloor n/(2w) \rfloor$, remove $U_{t,1}$
         from $U_A$ (i.e., $U_A = U_A \setminus U_{t,1}$), and set $\overline{D}_{t,1} = \emptyset$;
4      for each user $u_j \in U_{t,1}$ do
5          $\overline{v}_t^j \leftarrow LocSan(U_{t,1}, t, \epsilon, \Omega)$ via calling Algorithm 1;
6      end
7      Estimate $\overline{\mathbf{c}}_{t,1}$ from $\overline{D}_{t,1} = \{\overline{v}_t^j | u_j \in U_{t,1}\}$ with parameter $\epsilon$;
8      Calculate $dis = \frac{1}{d}\sum_{k=1}^{d}(\overline{\mathbf{c}}_{t,1}[k] - \mathbf{r}_{t-1}[k])^2 - \frac{1}{d}\sum_{k=1}^{d}\mathrm{Var}(\overline{\mathbf{c}}_{t,1}[k])$;
       // Sub Mechanism $\mathcal{M}_{t,2}$:
9      Calculate remaining population size $n_{rm} = n/2 - \sum_{i=t-w+1}^{t-1}|U_{i,2}|$;
10     Set number of potential publication users $n_{pp} = n_{rm}/2$;
11     Calculate potential publication error $err$ by Eq. (4) with $n = n_{pp}$ and $\epsilon$;
12     if $dis > err$ and $n_{pp} \geq u_{min}$ then
           // Publication Strategy
13         Sample a userset $U_{t,2}$ from $U_A$ with the size of $|U_{t,2}| = n_{pp}$,
             obtain $U_A = U_A \setminus U_{t,2}$, and set $\overline{D}_{t,2} = \emptyset$;
14         for each user $u_j \in U_{t,2}$ do
15             $\overline{v}_t^j \leftarrow LocSan(U_{t,2}, t, \epsilon, \Omega)$ via calling Algorithm 1;
16         end
17         Estimate $\overline{\mathbf{c}}_{t,2}$ from $\overline{D}_{t,2} = \{\overline{v}_t^j | u_j \in U_{t,2}\}$ with parameter $\epsilon$;
18         return $\mathbf{r}_t = \overline{\mathbf{c}}_{t,2}$;
19     else
           // Approximation Strategy
20         return $\mathbf{r}_t = \mathbf{r}_{t-1}$.
21     end
       // Recycling Users:
22     if $t \geq w$ then
23         $U_A = U_A \cup U_{t-w+1,1} \cup U_{t-w+1,2}$.
24     end
25 end
```

dissimilarity users over the w timestamps evenly. That is to say, at each timestamp t, $|U_{t,1}| = \lfloor n/(2w) \rfloor$ dissimilarity users report their value via an LDP mechanism using the entire budget ϵ.

Private strategy determination: In $\mathcal{M}_2$, the estimated dissimilarity dis (or approximation error) in $\mathcal{M}_1$ and the potential publication error err are compared to empirically choose a better strategy (i.e., with smaller error) from approximation and publication. err can be calculated based on the available budget ϵ and availabe user population n, e.g., according to Eq. (1) in GRR. Under the population division framework, the budget is fixed as a constant ϵ and err is mainly determined by the number of potential publication users, denoted as n_{pp}, which can be dynamically assigned in a sliding window. Since larger n_{pp} leads to less err, we should assign as many users as possible at each publication timestamp. However, since any user can only participate once in a window (ensuring w-event LDP), the allocation of n_{pp} at each timestamp and the scheduling of real publication users $U_{t,2}$ ($|U_{t,2}|$ should equal to n_{pp}) at publication timestamps should be carefully designed.

User allocation & recycling: In $\mathcal{M}_2$, with the above transition from budget division to population division, the adaptive budget allocation schemes, i.e., budget distribution (in LBD) and budget absorption (LBA) can be also borrowed for assigning the number of potential publication users n_{pp}. This also leads to two adaptive population division methods: *population distribution LPD* and *population absorption LPA*. Given n_{pp}, we can achieve maximized user utilization in scheduling publication users $U_{t,2}$ while ensuring w-event LDP for each user via a *user recylcing process*, which reuses the users assigned in timestamps out of the current sliding window.

5.2.2 *LDP Population Distribution (LPD).*

Algorithm 2 presents the details of LPD. Firstly, for dissimilarity dis calculation in $\mathcal{M}_1$ (Lines 3-8), the dissimilarity users U_1 is uniformly divided into w disjoint groups $U_{t,1}$ at each timestamp, i.e., $|U_{t,1}| = \lfloor n/(2w) \rfloor$. Next in $\mathcal{M}_2$, the remaining number of publication users n_{rm} is calculated by removing the already used publication users in the last $w - 1$ timestamps from the total number of publication users $n/2$ (Line 9). Then, the number of potential publication users is set as $n_{pp} = n_{\mathrm{rm}}/2$ to calculate a potential publication error err (Lines 10-11). By comparing err with dis, the publication or approximation strategy is decided then (Lines 12-21). In case of too many publications and n_{pp} decays too quickly to have no available user, a threshold $u_{\min}$ (e.g., $u_{\min} = 1$) is set (Line 12). Once publication is chosen, n_{pp} new users will be sampled as actual publication users $U_{t,2}$ from U_A to accomplish publication (Lines 13-18). Otherwise, $\mathbf{r}_t$ is approximated by $\mathbf{r}_{t-1}$, without using n_{pp} users (Line 20). Finally, both the used dissimilarity users and publication users (may be *null*) at timestamp $t - w + 1$, which is falling outside of the next active window, are recycled as available users U_A (Line 23). The recycling process ensures each user can contribute again after w timestamps while guaranteeing no users participate more than once.

5.2.3 *LDP Population Absorption (LPA).*

Algorithm 3 presents the details of LPA. The private dissimilarity calculation process of $\mathcal{M}_{t,1}$ in LPA is the same as that in LPD. In $\mathcal{M}_{t,2}$, the basic idea is to uniformly allocate users across timestamps then the unused publication users are absorbed at the timestamps where publication is chosen. Once a publication occurs at time l, the same number of users must be skipped from the succeeding timestamps to ensure available users within the active sliding window. So, the number of timestamps to be nullified t_N is first calculated based on the number of publication users at timestamp l, and thus skipped with approximation (Lines 4-6). After that, based on the timestamps can be absorbed, the number of potential publication users n_{pp} is calculated at each time t, which can further derive the potential publication error err (Lines 8-10). By comparing err with dis, $\mathcal{M}_{t,2}$ decides whether to freshly publish with the potential publication users (Lines 11-13) or continue to approximate with the last release (Lines 14-15). Similarly, both the used dissimilarity users and publication users at timestamp $t - w + 1$ are finally recycled as available users U_A, which is the same as LPD (Lines 18).

5.3 Analysis

5.3.1 *Privacy Analysis.*

LPD and LPA satisfy w-event LDP because each user reports to the server at most once in a time window of size w and each report goes through an ϵ-LDP protocol.

THEOREM 5.2. *LPD and LPA satisfies w-event LDP for each user.*

PROOF. See Appendix B.5 in [37]. □

5.3.2 *Utility Analysis.*

With the same assumptions, similar MSE expression can be obtained as Eq. (5) in Sec. 4.3.2. Then, in LPD, since the population is distributed to the m publications in an exponentially decreasing way, the population alloction sequence of $n_{t,2}$ is then $n/4, n/8, \ldots, n/2^{m+1}$. There is

$$\sum\nolimits_{i=1}^{m} \mathrm{Var}_{\mathsf{LPD}}[\mathbf{r}_{p_i}] = \sum\nolimits_{i=1}^{m} V(\epsilon, n/2^{i+1}) \tag{8}$$

Algorithm 3: LPA (at **Server-side**)

```
Input: Population $U$ of size $|U| = n$, budget $\epsilon$, window size $w$, domain $\Omega$
Output: Released statistics $R_t = (\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_t, \ldots)$
1  Initialize available userset $U_A = U$, and $\mathbf{r}_0 = \langle 0, \ldots, 0 \rangle$, last publication
     timestamp $l = 0$, and $\epsilon_{l,2} = 0$;
2  for each time $t$ do
       // Sub Mechanism $\mathcal{M}_{t,1}$:
3      Same as Lines 3-8 in Algorithm 2
       // Sub Mechanism $\mathcal{M}_{t,2}$:
4      Calculate timestamps to be nullified $t_N = \frac{|U_{l,2}|}{\lfloor n/(2w) \rfloor} - 1$;
5      if $t - l \le t_N$ then
6          return $\mathbf{r}_t = \mathbf{r}_{t-1}$;
7      else
8          Calculate timestamps can be absorbed $t_A = t - (l + t_N)$;
9          Set number of potential publication users
             $n_{pp} = \lfloor n/(2w) \rfloor \cdot \min(t_A, w)$;
10         Calculate potential publication error $err$ by Eq. (4);
11         if $dis > err$ then
               // Publication Strategy
12             Same as Lines 13-17 in Algorithm 2
13             return $\mathbf{r}_t = \bar{\mathbf{c}}_{t,2}$, set $l = t$;
14         else
               // Approximation Strategy
15             return $\mathbf{r}_t = \mathbf{r}_{t-1}$.
16         end
17     end
18     // Recycling Users:
19     Same as Lines 22-24 in Algorithm 2
20 end
```

Therefore, the error of LPD would still increase with m. However, according to Lemma 5.1, $V(\epsilon, n/2^{m+1}) < V(\epsilon/2^{m+1}, n)$. That is to say, LPD can achieve less error than LBD. Similarly, in LPA,

$$\sum\nolimits_{i=1}^{m} \mathrm{Var}_{\mathsf{LPA}}[\mathbf{r}_{p_i}] = m \cdot V(\epsilon, \frac{w+m}{4 \cdot w \cdot m} \cdot n) \tag{9}$$

which is smaller than $m \cdot V(\frac{w+m}{4 \cdot w \cdot m} \cdot \epsilon, n)$ in Eq. (7) of the budget absorption method LBA, given the same assumptions.

5.3.3 Communication Analysis. In LPD, all m publications in a window need $\sum_{i=1}^{m}(n/2^{i+1} + n/(2w)) = (\frac{1-(1/2)^m}{2} + \frac{m}{2w}) \cdot n$ users to report and the rest $w-m$ approximations need $\frac{w-m}{2w} \cdot n$ users. That is, for each window, there are $(\frac{1-(1/2)^m}{2} + \frac{m}{2w}) \cdot n + \frac{w-m}{2w} \cdot n = n(1 - \frac{1}{2^{m+1}})$ reports. Meanwhile, the server has to instruct the same number of users how to report. Therefore, the number of communication bits is at most $O(nd \cdot (\frac{1}{w} - \frac{1}{w \cdot 2^{m+1}}))$ from users to the server and $O(n \cdot (\frac{1}{w} - \frac{1}{w \cdot 2^{m+1}}))$ from the server to users per timestamp. Hence, the number of total communication bits per timestamp is at most $O(n \cdot (d+1) \cdot (\frac{1}{w} - \frac{1}{w \cdot 2^{m+1}}))$. In LPA, all m publications in a window need $m \cdot (\frac{w+m}{4 \cdot w \cdot m} \cdot n + n/(2w))$ users to communicate and the rest timestamps need $(w-m) \cdot n/(2w)$ users. Similar to LPD, the number of communication bits between the users and server per timestamp is at most $O(n \cdot (d+1) \cdot \frac{1}{w}[m \cdot (\frac{w+m}{4 \cdot w \cdot m} \cdot n + n/(2w)) + (w-m) \cdot n/(2w)]) = O(n \cdot (d+1) \cdot (\frac{1}{2w} + \frac{w+m}{4w^2}))$.

6 EXTENSIONS AND DISCUSSIONS

In this section, we extend our proposed framework of online adaptive population division and recycling to different analytic tasks, settings and other existing stream algorithms.

6.1 Extension to Other Tasks and Settings

6.1.1 Mean Estimation for Numerical Domain. Besides frequency estimation, mean estimation over numerical domain is also a typical analytic task in LDP. We here adopt the Hybrid Mechanism as the primitive mean estimation protocol to briefly discuss the extension.

HM. Hybrid Mechanism (HM) [42] is an LDP mean estimation mechanism, which combines the advantages of Stochastic Rounding (SR) [15] and Piecewise Mechanism (PM) [42] with the minimum error. When $\epsilon > 0.61$, users use PM (which has the variance of $\frac{v^2}{e^{\epsilon/2}-1} + \frac{e^{\epsilon/2}+3}{3(e^{\epsilon/2}-1)^2}$) with probability $1 - e^{-\epsilon/2}$ and SR (which has the variance of $(\frac{e^{\epsilon}+1}{e^{\epsilon}-1})^2 - v^2$) with probability $e^{-\epsilon/2}$. When $\epsilon \le 0.61$, only SR will be called. Considering most private values $v \in [-1, 1]$ are close to 0 in practice, the worst-case variance of the perturbed value $\bar{v}$ in HM is written as

$$\mathrm{Var}^*[\bar{v}] = \begin{cases} \left(\frac{e^{\epsilon}+1}{e^{\epsilon}-1}\right)^2, & \text{if } \epsilon \le 0.61, \\ \frac{1}{e^{\epsilon/2}}\left[\left(\frac{e^{\epsilon}+1}{e^{\epsilon}-1}\right)^2 + \frac{e^{\epsilon/2}+3}{3(e^{\epsilon/2}-1)^2}\right], & \text{if } \epsilon > 0.61. \end{cases} \tag{10}$$

The variance of the mean estimate $\bar{c} = \frac{1}{n}\sum_{j=1}^{n} \bar{v}^j$ ($\bar{v}^j$ denotes the output of j-th user), can be computed as $\mathrm{Var}[\bar{c}] = \frac{1}{n^2}\sum_{j=1}^{n} \mathrm{Var}[\bar{v}^j]$. Such a variance calculation contains true values v^j, we can simply approximate it as $\mathrm{Var}[\bar{c}] = \frac{1}{n}\mathrm{Var}^*[\bar{v}]$. Then, the above HM mechanism can be embedded to the LDP-IDS framework for mean estimation over infinite streams with LDP. Particularly, on the users' side, each user adopts HM at each timestamp and reports to the server. Receiving the reports, the server estimates a dissimilarity dis over the last released mean value, compares it with the potential error err, and determines whether to perturb or approximate. Both adaptive budget and population division methods can be developed.

6.1.2 Dealing with Large Domain. For real-world streams, the domain is often large in size d (categorical data) or has a large numerical bound B for a mapped range $[0, B]$ (numerical data). For categorical data, we can properly bin the domain to limit its size. For continuous domains, it is also effective to clip the bound as the upper bound can be rarely large but most data items are far smaller than it. One straightforward method is to simply truncate the domain by a fixed threshold (e.g., using the 99.5-th percentile [34]). The state-of-the-art method ToPL [45] caches a period of reports to privately find the optimal threshold by minimizing the overall estimation errors, and then uses this threshold in the later sequence. This idea can be borrowed in our algorithms for dealing with large domains. Note that, we are using a flat setting, which is different from the hierarchical one in [45]. Given a threshold θ, the expected square error of mean estimation should be simply modeled as $\mathrm{Var}(\bar{\mathbf{c}}) + (\sum_{\theta<k<B} \bar{f}_k \cdot (k - \theta))^2$ where $\mathrm{Var}(\bar{\mathbf{c}})$ denotes the variance of the estimated statistic $\bar{\mathbf{c}}$ and $\bar{f}_k$ denotes the estimated frequency of value k. To estimate an optimal θ, the expression can be minimized from a period of cached reports.

6.2 Extension to Other Stream Methods

Our LDP-IDS framework can be also extended to various existing streaming algorithms, including event-level private PeGaSus [7], user-level private FAST [23], and w-event private algorithms DSAT$_w$ [30] and RescueDP [43], with consideration of LDP error analysis and some additional efforts in parameter adaptation. Due to space limitations, their details are described in [37]. Here, we briefly discuss the high level ideas of adapting DSAT$_w$ and RescueDP.

Algorithm 4: LP-DSAT$_w$

```
Input: Population $U$ of size $n$, budget $\epsilon$, window size $w$, cutoff point $C$, PID
       controller parameters $\theta$, parameter $\sigma$, prior threshold $T$
Output: Released statistics $R_t = (\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_t, \ldots)$
1  Initialize available userset $U_A = U$, $n_1 = \beta n$, and $n_2 = (1-\beta)n$, $\tilde{T}_1 = T$;
2  for each timestamp $t$ do
3      $n_{rm} = n_2 - \sum_{i=t-w+1}^{t-1} |U_{i,2}|$;
4      if $n_{rm} \leq 0$ then
5          return $\mathbf{r}_t = \mathbf{r}_{t-1}$;
6      else
           // Distance computation:
7          Sample users $U_{t,1}$ from $U_A$ with the size of $\lfloor n_1/C \rfloor$, remove $U_{t,1}$
             from $U_A$, i.e., $U_A = U_A \setminus U_{t,1}$, set $\overline{D}_{t,1} = \emptyset$;
8          Same as Lines 4-8 in Algorithm 2
9          Compute $count = \frac{\sum_{i=t-w+1}^{t-1} |U_{i,2}|}{\lfloor n_2/C \rfloor}$;
           // PID Controller-based Adaptive Threshold Control:
10         Compute $E_t = \left|\frac{count}{t} - \frac{C}{w}\right|$, $e_t = \frac{|E_t - \sigma|}{\sigma}$, $u_t = \mathrm{PID}(e_t; \theta)$;
11         if $\frac{count}{t} - \frac{C}{w} < 0$ then
12             $\tilde{T}_t = \max\{0, \tilde{T}_t - u_t\}$;
13         else
14             $\tilde{T}_t = \min\{2, \tilde{T}_t + u_t\}$
15         end
16         if $dis \geq \tilde{T}_t$ then
               // Perturbation
17             Sample users $U_{t,2}$ from $U_A$ with the size of $\lfloor n_2/C \rfloor$, remove
                 $U_{t,2}$ from $U_A$, i.e., $U_A = U_A \setminus U_{t,2}$, set $\overline{D}_{t,2} = \emptyset$;
18             Same as Lines 4-8 in Algorithm 2
19             $count = count + 1$, $l = t$;
20             return $r_t = \overline{c}_{t,2}$.
21         else
22             return $r_t = r_{t-1}$.
23         end
24     end
25     Same as Lines 22-24 in Algorithm 2
26 end
```

Algorithm 5: LP-RescueDP

```
Input: Population $U$ of size $n$, budget $\epsilon$, window size $w$, maximum user
       allocation $n_{max}$, maximum allocation portion $p_{max}$, scale factor $\phi$
Output: Released statistics $R_t = (\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_t, \ldots)$
1  Initialize available userset $U_A = U$;
2  for each timestamp $t$ do
3      Obtain a prior estimate $\tilde{\mathbf{c}}_t$ via KF-Prediction;
4      if $t$ is not a sampling timestamp then
5          return $\mathbf{r}_t = \mathbf{r}_{t-1}$
6      else
           // Adaptive Population Allocation
7          Calculate the number of available users $n_A = n - \sum_{k=t-w+1}^{t-1} n_k$;
8          Obtain the portion $p$ =min$(\phi \cdot \ln(I+1), p_{max})$ and the number
             of allocated users $n_t$ =min$(\lfloor p \cdot n_A \rfloor, n_{max})$;
           // LDP Perturbation
9          Sample a userset $U_t$ from $U_A$ with the size of $n_t$, $U_A = U_A - U_t$,
             set $\overline{D}_t = \emptyset$;
10         for each user $u_j \in U_t$ do
11             $\overline{v}_t^j \leftarrow LocSan(U_t, t, \epsilon, \Omega)$ via calling Algorithm 1;
12         end
13         Estimate $\overline{\mathbf{c}}_t$ from $\overline{D}_t = \{\overline{v}_t^j | u_j \in U_t\}$ with parameter $\epsilon$;
           // Kalman Filtering
14         Obtain a posterior estimate $\hat{\mathbf{c}}_t$ via KF-Correction;
           // Adaptive Sampling
15         Calculate the feedback error $E_t = |\hat{\mathbf{c}}_t - \mathbf{r}_l|$ and PID error $\delta$;
16         Determine the new sampling interval
             $I_t = \max([1, I_l + \theta(1 - (\delta/\mathrm{Var}[\overline{\mathbf{c}}_t])^2)])$ and set $l = t$;
17         return $\mathbf{r}_t = \hat{\mathbf{c}}_t$.
18     end
19     if $t \geq w$ then
           // Recycling Users
20         $U_A = U_A \cup U_{t-w+1}$;
21     end
22 end
```

DSAT$_w$ is a w-event private CDP algorithm for continuous histogram release [30]. It also adopts the framework of perturbation or approximation but assumes that there are totally $C < w$ sampling timestamps in a time window of size w. In DSAT$_w$, the dissimilarity is also computed, but compared with a changeable threshold $\tilde{T}_t$. The intuition is, if there are many sampling (i.e., perturbation) timestamps, the threshold is increased to reduce sampling frequency; otherwise, it is decreased. Based on our proposed LDP-IDS framework, both budget and population division-based LDP variants (denoted as LB-DSAT$_w$ and LP-DSAT$_w$ respectively) of DSAT$_w$ can be derived. We briefly discuss LP-DSAT$_w$ (shown in Algorithm 4) as follows. Initially, the total population is divided into dissimilarity population and publication population, which are set as n_1 and n_2, respectively. And the initial threshold $\tilde{T}_1 = T$ is set based on some prior knowledge. At each time t, LP-DSAT$_w$ first calculates the remaining publication population n_{rm} (Line 3). If no publication population left, it directly approximates as the last release (Lines 4-5). Otherwise, it samples a user set $U_{t,1}$ of population $\lfloor n_1/C \rfloor$ from U_A to report values with ϵ-LDP, from which, a dissimilarity (distance) dis can be derived (Lines 6-8). Then, the number of published timestamps is counted according to the used publication population (Line 9). Based on the difference E_t between current sampling frequency $count/t$ and the expected one C/w, a PID error e_t is fed into a PID controller with parameter θ to compute an update u_t (Line 10). The threshold $\tilde{T}_t$ is then increased or decreased accordingly (Lines 11-15). After that, dis is compared with $\tilde{T}_t$ to choose between publication and approximation. If a publication occurs, a new user set $U_{t,2}$ of population $\lfloor n_2/C \rfloor$ is sampled to report with ϵ-LDP (Lines 16-18). Similar to LPD/LPA, the used users $U_{t-w+1,1}$ and $U_{t-w+1,2}$ at timestamp $t-w+1$ are recycled (Line 25).

RescueDP [43] represents a w-event private mechanism for real-time spatio-temporal data release. It consists of several components: dynamic grouping, adaptive sampling, Kalman filter, and adaptive budget allocation. The dynamic grouping strategy deals with high dimensional streams, which is beyond the scope of this paper. The adaptive sampling strategy leverages a PID controller to dynamically adjust the sampling frequency. The filter component assumes the perturbation error in stream approximately follows a Gaussian distribution $\mathcal{N}(0, R)$ and exploits a Kalman filter to minimize noise in real-time. The adaptive budget allocation assigns budget in a logarithmic decaying way while satisfying w-event privacy. As a budget division CDP algorithm, RescueDP, can be easily transferred to both budget and population division-based LDP variants (denoted as LB-RescueDP and LP-RescueDP respectively). We mainly describe the latter (shown in Algorithm 5). At each timestamp, it first obtains a prior estimate $\tilde{\mathbf{c}}_t$ (e.g., approximating as the last release) (Line 3). If it is not a sampling timestamp, $\tilde{\mathbf{c}}_t$ is directly released (Lines 4-5); otherwise, some users are sampled to report and derive a fresh estimate with ϵ-LDP. Particularly, it first calculates the remaining available population $n_r = n - \sum_{k=t-w+1}^{t-1} n_k$ (Line 7). Then, it assigns the remaining population to the sampling timestamps in a logarithmic decaying way, meanwhile limiting the maximum population allocated at each time (Line 8). With the assigned population n_t, LP-RescueDP randomly samples a user set from U_A to report values with ϵ-LDP, from which, it can derive an estimate $\overline{\mathbf{c}}_t$ (Lines 9-13). Next, the Kalman filter produces a posterior estimate via combining the prior estimate $\tilde{\mathbf{c}}_t$ and the derived $\overline{\mathbf{c}}_t$ (Line 14). To make Kalman filter effectively eliminate noise in the stream, the parameter R can be empirically set as the populatio division-based estimate variance $R \propto V(\epsilon, n/m)$. The sampling interval is dynamically adjusted according to the distance between the released results $E_t = |\mathbf{r}_t - \mathbf{r}_l|$, which is transformed as a PID error δ for computing the new sampling interval (Lines 15-16).

7 PERFORMANCE EVALUATION

In this section, we conducted extensive experiments to evaluate the performance of our proposed framework and algorithms.

7.1 Experimental Setup

7.1.1 ***Synthetic Datasets.*** We synthesized binary streams with different sequence models. Given a probability process model $p_t = f(t)$, the length of time T, and user population n, we first generated a probability sequence $(p_1, p_2, \ldots, p_T)$ with T timestamps. Then, at each timestamp t, we randomly chose a portion of p_t users from the total n users to set their true report value v_t^j as 1, and the rest as 0. The following typical sequence patterns were used.

LNS is a linear process $p_t = p_{t-1} + \mathcal{N}(0, Q)$, where $p_0 = 0.05$ and $\mathcal{N}(0, Q)$ is Gaussian noise with the standard variance $\sqrt{Q} = 0.0025$. **Sin** is a sequence composed by a sine curve $p_t = Asin(bt) + h$ with $A = 0.05$, $b = 0.01$ and $h = 0.075$. **Log** is a series with the logistic model $p_t = A/(1 + e^{-bt})$ where $A = 0.25$ and $b = 0.01$. For mean estimation over continuous domain, we further synthesized the real-valued versions of above datasets (denoted as **LNS-Con**, **Sin-Con**, and **Log-Con** respectively) by sampling random numbers following a Gassuian distribution $\mathcal{N}(p_t, \sigma^2)$ with the mean of p_t in above discrete data generation and standard variance of $\sigma = 10$.

Without specifying, we used above models and default parameters to generate synthetic binary streams with 800 timestamps of 200, 000 users. To demonstrate the varying fluctuations, we set n fixed but changed the parameters Q in LNS and b in Sin respectively to obtain different datasets. To demonstrate the varying populations, we used the probability sequences generated with the default parameters above, but performed different number of sampling processes to obtain datasets with different population n.

7.1.2 ***Real-world Datasets.*** To evaluate practical performance, the following non-binary real-world datasets were also used.

For frequency estimation over categorical domains, **Taxi**[3] contains the taxi trajectories during the period of Feb. 2 to Feb. 8, 2008 within Beijing. We obtain $N = 10,357$ data streams for each taxi by extracting $T = 886$ timestamps (each at 10-minute level) and partitioning area into 5 grids, i.e., $d = 5$. **Foursquare**[4] includes check-ins of Foursquare users from Apr. 2012 to Sep. 2013 over 77 countries, each record including time, place and user ID. We transform it into $N = 266,909$ data streams with the length of $T = 447$ timestamps, each represents a user's check-in sequence over $d = 77$ countries. **Taobao**[5] contains the AD click logs of 1.14 million customers and a total of 12,973 categorizes at Taobao. For simplicity, we grouped the AD commodities into $d = 117$ categorizes. Then, we extracted $N = 1,023,154$ customers' click streams, where each item corresponds to the categorize of the user's last click during each minute in one day, i.e., $T = 1440$ timestamps. For mean estimation over numerical domains, we obtained the labeled price of each AD commodities in Taobao, which is referred to as **Taobao-Price**. It has the maximal value of 1, 188, 429 and the percentiles $p_{99.9} = 2,232.54$, $p_{99.5} = 650.44$, $p_{99} = 411.79$, and $p_{98} = 189.90$.

7.1.3 ***Compared Algorithms.*** We compared the following algorithms. All are implemented using Matlab 2020.

Framework	Non-adaptive allocation		Adaptive allocation	
	Uniform	Sampling	Distribution	Absorption
Budget	LBU	LSP	LBD	LBA
Population	LPU		LPD (Algorithm 2)	LPA (Algorithm 3)

LBU (Sec. 4.1) and LPU (Sec. 5.1) are baseline methods that uniformly divide budget and population, respectively. LSP (Sec. 4.1 and 5.1) invests the entire budget and users at sampling timestamps with fixed interval. LBD/ LBA (Sec. 4.2.3) and LPD/LPA (Secs. 5.2.2 and 5.2.3) adaptively allocate budget and population via two different schemes. In above algorithms, besides frequency estimation over categorical domains, we also validated the mean estimation mechanism HM for numerical domains. Finally, we implemented the prototypes of other stream algorithms discussed in Sec. 6.2.

All experiments were conducted on a PC with an Intel Core i5-6300HQ 3.20GHz and 16GB memory.

7.1.4 ***Performance Metrics.*** We evaluated the algorithm performance in terms of data utility, event monitoring efficiency, and communication efficiency. The utility was measured as the *mean relative error* (MRE) between the released and true statistics. The event monitoring efficiency was measured as the ratio that, from perturbed reports, the server successfully detects extreme events, i.e., the statistics of which are greater than a given threshold. Considering different stream length, communication efficiency is measured by the time average number of communication bits per user.

7.2 Experimental Results

7.2.1 ***Overall Utility.*** Fig. 3 shows the accuracy of all w-event LDP methods on all datasets, with different ϵ. Overall, the error of all methods decreases with ϵ, which shows the tradeoff between data utility and privacy. Besides, the population division-based methods significantly outperform budget-division ones with much smaller MRE. LBD/LBA generally shows smaller error than the straightforward method of LBU. This is because LBD/LBA can utilize temporal correlations in data streams to reduce the privacy budget consumption rate. The advantage of LPD/LPA is clearer as noise does not increase dramatically when the population is divided. Although LSP achieves even smaller error than LPD/LPA, its performance varies dramatically across datasets.

Fig. 4 shows the accuracy of all w-event LDP methods, with different window size w, on all datasets. The MRE of all methods increases with w, since less budget or users will be allocated to each timestamp. For budget division methods, with the increasing w, LBD distributes budget in an exponentially decaying way and allocates very small budget for the newest timestamp in the window, thus causing large estimation error. LBA avoids this issue and well adapts to data fluctuations. For population division methods, despite the similar trends, LPD manages to achieve smaller MRE than LPU

[3]https://www.microsoft.com/en-us/research/publication/t-drive-trajectory-data-sample/
[4]https://sites.google.com/site/yangdingqi/home/foursquare-dataset
[5]https://tianchi.aliyun.com/dataset/dataDetail?dataId=56

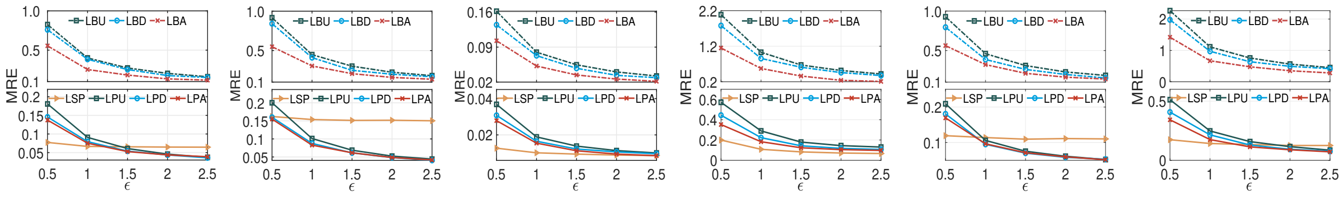

(a) LNS (b) Sin (c) Log (d) Taxi (e) Foursquare (f) Taobao

Figure 3: Data utility with different ϵ ($w = 20$)

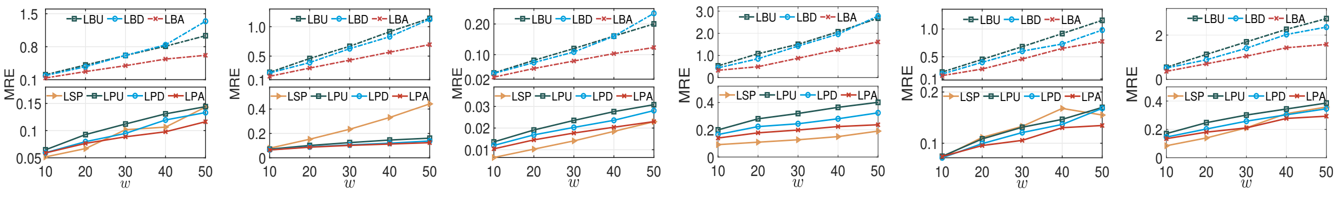

(a) LNS (b) Sin (c) Log (d) Taxi (e) Foursquare (f) Taobao

Figure 4: Data utility with different w ($\epsilon = 1$)

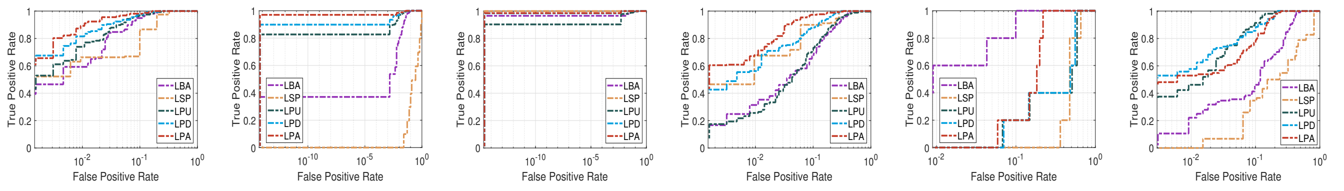

(a) LNS (b) Sin (c) Log (d) Taxi (e) Foursquare (f) Taobao

Figure 5: ROC Curve for Event Monitoring ($\epsilon = 1$, $w = 50$)

while LPA performs even much better. Besides, as shown, with large w, LPD/LPA gains more prominent advantages.

7.2.2 ***Event Monitoring Efficiency.*** Besides the overall distance metric of MRE, event monitoring is commonly used in streams to detect whether the estimate at each timestamp is larger than a given threshold δ. Fig. 5 displays the ROC curves for detecting above-threshold points on all six dataset. On synthetic binary datasets LNS, Sin, Log, δ was directly set as $0.75 \times (\max(\mathbf{c}) - \min(\mathbf{c})) + \min(\mathbf{c})$. On other three non-binary real-world datasets, we monitored the mean-value $\mathbf{c}_{\text{mean}}$ of the histogram $\boldsymbol{c}_t$ and δ was set as $0.75 \times (\max(\mathbf{c}_{\text{mean}}) - \min(\mathbf{c}_{\text{mean}})) + \min(\mathbf{c}_{\text{mean}})$. Overall, the population division methods perform better than the budget division method LBA, as they can achieve higher accuracy in above-threshold value detection. Despite varying on different datasets, LPD and LPA in general outperform the other three methods. Although LSP manages to have much smaller MRE in Figs. 3 and 4, it generally performs the worst on most datasets as too many approximations hinder its efficiency in detecting real-time changes.

7.2.3 ***Effect of Dataset Parameters.*** To demonstrate the impact of dataset parameters on utility, we changed population n, and noise variance Q and period parameter b in the synthetic datasets LNS and Sin respectively. Note that, while varying the population size n, we kept the frequency fixed. As shown in Figs. 6(a) and 6(b), the MRE of all methods decreases with n. This is because that, enlarging n while fixing the frequency leads to better estimation accuracy. Figs. 6(c) and 6(d) show the MRE on LNS with different Q and Sin with different b, respectively. On LNS, the MRE of all methods increases with Q, which measures the fluctuation in streams. This result verifies that these data-dependent methods perform better on streams with few changes. Although LSP manages to have the smallest error when the variance is small (i.e., $\sqrt{Q} = 0.001, 0.002$), it grows fast and surpasses LPD and LPA with the increase of Q. LPD and LPA induce much smaller error, and perform slightly worse than LPU when the variance is large. Note that, b also represents data fluctuations and larger b means larger fluctuations. Similar conclusions can be obtained from the results on Sin.

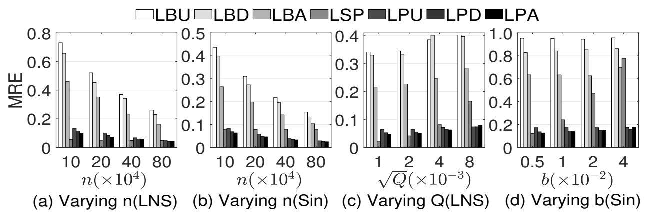

(a) Varying n(LNS) (b) Varying n(Sin) (c) Varying Q(LNS) (d) Varying b(Sin)

Figure 6: Effect of parameters n, Q, b on MRE ($\epsilon = 1$, $w = 20$)

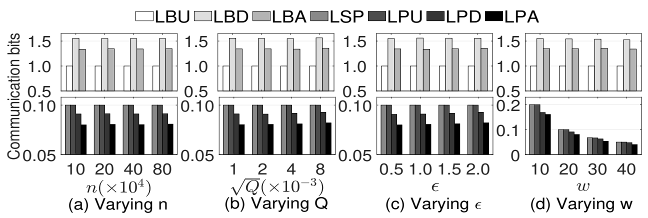

(a) Varying n (b) Varying Q (c) Varying ϵ (d) Varying w

Figure 7: Average #. of Communication Bits Per User (LNS)

7.2.4 ***Communication Efficiency.*** Fig. 7 compares the average number of communication bits per user in different methods, with different parameters on LNS. As shown, communication bits in population division methods are significantly less than those in budget division methods, since only a small portion of users contribute data at each timestamp. Fig. 7(a) depicts the impact of population n. For both adaptive methods LBD/LBA and LPD/LPA, the communication bits become slightly less as n increases. This is because that, in the synthesized dataset with the same distribution, with the increase of n, publication will incur more LDP noise and occur infrequently. Fig. 7(b) shows the impact of data variance Q. Similarly, Q has no impact on data-independent methods LBU and LSP. However, with the increase of Q, data-dependent methods LBD/LBA and LPD/LPA incur slightly more bits as they would increase publication frequency streams with more fluctuations. Fig. 7(c) presents the communication bits wrt. ϵ. Budget division methods nearly keep the same as a slight increase of ϵ has limited impact. However, population division methods increase with ϵ since the publication error would become smaller and more publications would be chosen in the adaptive methods. Fig. 7(d) shows the communication bits wrt. w. LBU keeps unchanged while LBD/LBA shows a slight decrease. This is because the publication noise increases with w and approximation is more favorable. All population methods decrease with w because the sampling ratio of each user drops with w.

Table 2 further compares the average #. of communication bits per user among different methods and datasets. The population division methods manage to have much less communication overhead. Data-adaptive methods LPD and LPA further reduce the communication cost via exploiting the sparsity in data streams. The scale of bit number is different among datasets as different frenquency oracles are adaptively chosen based on parameters d and ϵ. All above results are consistent with the analysis in Sec. 4.3.3 and 5.3.3.

Table 2: Time Average #. of Communication Bits Per User

$\epsilon = 1, w = 20$		Synthetic			Real-world		
		LNS	**Sinusoidal**	**Logistic**	**Taxi**	**Foursquare**	**Taobao**
B	**LBU**	1.0000	1.0000	1.0000	3.0000	77.0000	118.0000
	LBD	1.5585	1.5516	1.5572	4.2195	102.5540	146.0972
	LBA	1.3410	1.3380	1.3284	3.8126	93.4408	136.9553
P	**LSP**	0.1000	0.1000	0.1000	0.2032	4.0208	6.0602
	LPU	0.1000	0.1000	0.1000	0.1998	3.9000	5.9496
	LPD	0.0912	0.0913	0.0915	0.1870	3.7815	5.5106
	LPA	0.0804	0.0806	0.0803	0.1651	3.5701	4.7819
$\epsilon = 2, w = 20$		**Synthetic**			**Real-world**		
		LNS	**Sinusoidal**	**Logistic**	**Taxi**	**Foursquare**	**Taobao**
B	**LBU**	1.0000	1.0000	1.0000	3.0000	77.0000	118.0000
	LBD	1.5500	1.5410	1.5540	4.2810	102.9946	144.9437
	LBA	1.3540	1.3580	1.3368	3.8030	93.3017	136.7143
P	**LSP**	0.1000	0.1000	0.1000	0.2032	3.9431	6.0602
	LPU	0.1000	0.1000	0.1000	0.1998	3.9000	5.9496
	LPD	0.0934	0.0933	0.0937	0.1917	3.8728	5.5243
	LPA	0.0825	0.0828	0.0822	0.1746	3.8277	4.7802
$\epsilon = 2, w = 40$		**Synthetic**			**Real-world**		
		LNS	**Sinusoidal**	**Logistic**	**Taxi**	**Foursquare**	**Taobao**
B	**LBU**	1.0000	1.0000	1.0000	3.0000	77.0000	118.0000
	LBD	1.5420	1.5268	1.4832	4.1953	99.4699	145.3741
	LBA	1.3395	1.3656	1.3148	3.7991	92.6292	136.4044
P	**LSP**	0.0500	0.0500	0.0500	0.1038	2.0048	3.0301
	LPU	0.0500	0.0500	0.0500	0.0999	1.9500	2.9748
	LPD	0.0485	0.0484	0.0490	0.0987	1.9454	2.8806
	LPA	0.0410	0.0411	0.0414	0.0863	1.8943	2.3986

B and **P** refer to budget division and population division, respectively.

7.2.5 *Varying Budget Portion.*

Fig. 8 shows the performance of data-adaptive methods when shifting the budget between two sub-mechanisms. Specifically, we set β portion of budget or population for $\mathcal{M}_{i,1}$ and the rest $1-\beta$ available for $\mathcal{M}_{i,2}$. Note that, for simplicity, we set $\beta = 0.5$ as the default value in our algorithm description. In general, LBD and LPD have a concave pattern when varying β from 0.1 to 0.9, albeit having different minimum point. Roughly, in most cases, they have the minimal error around $\beta = 0.5$. LBA and LPA generally have an increasing trend with β. This is because that, with the increase of β, there will be many nullified timestamps in $\mathcal{M}_{i,2}$, which causes the waste of budget or population invested in $\mathcal{M}_{i,1}$. For smaller error, we can set a small β. However, as shown, the error is still much close to that around $\beta = 0.5$.

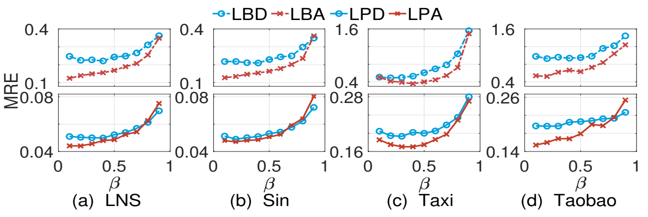


Figure 8: Varying budget allocation portion β ($\epsilon = 1$, $w = 20$)

7.2.6 *Mean Estimation over Numerical Domain.*

We next validate the algorithm performance, particularly mean estimation, on numerical domains. We used HM as the primitive LDP protocol. Considering the large but sparse numerical domains, Fig. 9 first shows the impact of domain truncation on dataset Taobao-price with different thresholds, which are represented by different percentiles. As shown, without clipping ($\theta = p_{100}$), all the population division algorithms have extravagant error on mean estimation. With the decrease of the clipping threshold, i.e. from p_{100} to p_{98}, the estimation error can be reduced by a factor of 4 orders of magnitude. However, as shown in Fig. 9(d), for larger ϵ (e.g., $\epsilon \geq 1.0$), the estimation error ($0.74 \sim 0.76$) becomes larger than those ($0.6 \sim 0.7$) in Fig. 9(c). This illustrates that, with further reduced threshold in domain truncation, the estimation error becomes larger again and harms the utility. And, there exists an optimal threshold.

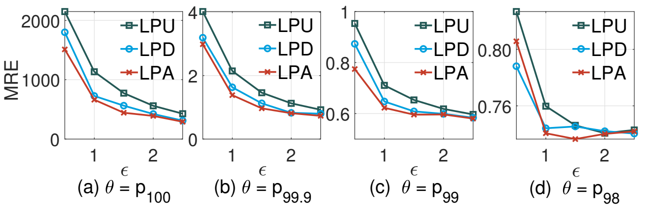


Figure 9: Impact of Threshold θ (Taobao-Price, $w = 20$)

Fig. 10 demonstrates the impact of parameters on mean estimation accuracy. Figs. 10(a) and 10(b) show the impact of data fluctuations on two synthetic datasets LNS-Con and Sin-Con with small domains. Both figures show very similar trends as Figs. 6(c) and 6(d). Figs. 10(c) and 10(d) show the mean estimation accuracy on dataset Taobao-Price with a large domain size. Particularly, we borrowed the idea of optimal threshold clipping in [45] and truncated its domain first. Since domain truncation is orthogonal to our online adaptive methods, the trends are similar to those in the frequency estimation and the population division methods still show significantly less error than the budget division ones.

7.2.7 *Extension to Other Stream Algorithms.*

Fig. 11 demonstrates the results of extending other existing online adaptive algorithms in our LDP-IDS framework. As discussed in Sec. 6.2, these algorithms are all adapted into the w-event LDP setting via both budget division (denoted as LB-*) and population division (denoted

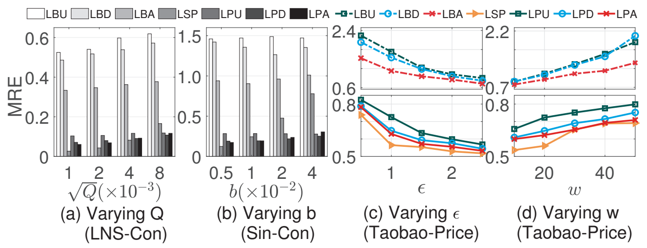


Figure 10: Mean Estimation (threshold truncation, $\epsilon = 1$, $w = 20$)

as LP-*). The detailed algorithm descriptions and their parameter setups are shown in Appendix C and D.1 in [37] respectively. Note that, these algorithms originally focus on slightly different aggregates (e.g., histogram in DSAT while count in the rest). For simple and fair comparison, these extended LDP algorithms are mainly evaluated based on frequency estimation over binary streams. Both Taxi and Taobao are binarized to include real-word stream patterns. As shown, all the population division versions can significantly reduce the overall error, compared to the budget division versions. Compared to the proposed algorithms LBA/LPA (without manually set parameters), these adapted algorithms may achieve higher accuracy with some parameter tuning. Nonetheless, in the population division methods, the differences are much smaller.

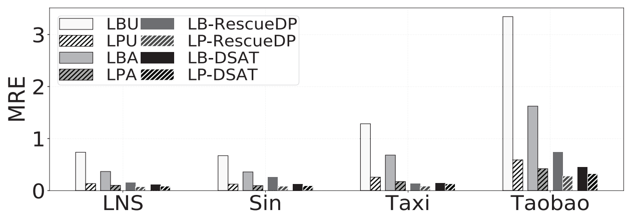


Figure 11: Performance of All Stream Algorithms ($\epsilon = 1$, $w = 30$)

7.2.8 ***Summary.*** We briefly summarize the above results as follows. (1) The analytic tasks (frequency and mean estimation) are orthogonal to and can be adapted into the stream algorithms in this paper. (2) Overall, population division methods (LPD/LPA, LSP, LPU), always perform much better than their budget division competitors (LBD/LBA, LBU) in the LDP setting. (3) In real-time event detection scenarios, both data-adaptive population division methods are superior than sampling-based (LSP) and uniform methods (LPU). (4) For larger window w or smaller budget ϵ, it is better to choose absorption methods (LBA/LPA) than distribution methods (LBD/LPD). (5) For streams with small fluctuations, data-adaptive methods (LPD/LPA, LBD/LBA) are better than sampling (LSP) and uniform methods (LPU, LBU). (6) The population division framework has great compatibility for various LDP protocols and flexibility of incorporating ideas in existing stream algorithms.

8 RELATED WORK

Centralized Differential Privacy (CDP) [16] on streams originally focuses on two notions: event-level DP [5–7, 18] and user-level DP [18, 23]. The former is usually insufficient for privacy protection while the latter can only be achieved on a finite stream [23]. To address the dilemma, [29] proposes a notion of w-event DP for infinite streams, which ensures ϵ-DP for any time window including w consecutive timestamps. Based on a *sliding window* methodology, they further propose two methods satisfying w-event privacy, *Budget Distribution* (BD) and *Budget Absorption* (BA) to effectively allocate privacy budget considering that the statistics on streams may not change significantly in successive timestamps [43].

LDP also suffers from low utility in the streaming setting [3, 40]. To this end, early studies adopt a simple memoization step to provide longitudinal LDP guarantee [2, 3, 21]. Later, many CDP algorithms have been extended to the LDP setting. [26] and [45] focus on event-level LDP for data streams. [4] proposes an (ϵ, δ) user-level LDP algorithm for finite streaming data collection using the analytic Gaussian mechanism, which focuses on approximate DP and has to renew privacy budget periodically. [48] proposes a pattern-aware stream data collection mechanism with a metric based w-event LDP, which is not directly comparable to our work. More importantly, all these approaches enforce LDP over streams via traditional budget division methodology in CDP, which causes severe utility loss as reporting with low LDP budget is rather noisy.

Recently, several LDP studies [9, 33, 44] have shown that one can partition users to answer multiple questions with LDP, which still satisfy the same LDP under the parallel composition but can achieve much higher accuracy than splitting privacy budget and adopting the sequential composition. In the finite study [20], each user randomly reports with the entire budget on the nodes at a fixed level of the binary tree. However, users are allocated in advance for a fixed time interval but not on-the-fly for infinite streams. Despite these pioneering studies, the idea of population division cannot be directly extended to infinite streams with LDP. Detailed related work can be referred to Appendix A in [37].

9 CONCLUSION

We propose LDP-IDS, a locally privacy-preserving paradigm for infinite streaming data collection and analysis. We first formalize the problem of w-event LDP for infinite streams and formulate the budget division methodology. By constructing a unified error analysis in LDP, we present two adaptive budget division methods for LDP-IDS with enhanced utility, which considers the non-deterministic sparsity in streams. More importantly, we propose a novel population division and recycling framework for LDP-IDS, and two corresponding data-adaptive population division methods, which can achieve significant utility improvement and communication reduction. In addition, we extend our proposed framework to different analytic tasks, settings and other existing stream algorithms. Through extensive theoretical analysis and experiments with real-world datasets, we demonstrate that our proposed framework and methods not only have superior performance in terms of data utility, practical event monitoring and communication efficiency, but also enjoy great compatibility and flexibility.

ACKNOWLEDGMENTS

This work was supported in part by the National Key Research and Development Program of China under Grant 2020YFA0713900; in part by the National Natural Science Foundation of China under Grants 61772410, 61802298, 61972203, 62172329, U1811461, U21A6005, and 11690011; in part by the China Postdoctoral Science Foundation under Grant 2020T130513 and 2019M663726; and in part by Natural Science Foundation of Jiangsu Province under Grant BK20190442.

REFERENCES

[1] H. Arcolezi, J. Couchot, B. Al Bouna, and X. Xiao. 2021. Random Sampling Plus Fake Data: Multidimensional Frequency Estimates With Local Differential Privacy. In *Proc. ACM CIKM.* 47–57.

[2] H. Arcolezi, J. Couchot, B. Bouna, and X. Xiao. 2021. Improving the Utility of Locally Differentially Private Protocols for Longitudinal and Multidimensional Frequency Estimates. *arXiv preprint arXiv:2111.04636* (2021).

[3] H. Arcolezi, J Couchot, B. Bouna, and X. Xiao. 2021. Longitudinal Collection and Analysis of Mobile Phone Data with Local Differential Privacy. *Privacy and Identity Management: 15th IFIP WG 9.2, 9.6/11.7, 11.6/SIG 9.2. 2 International Summer School, Maribor, Slovenia, September 21–23, 2020, Revised Selected Papers* 619 (2021), 40.

[4] E. Bao, Y. Yang, X. Xiao, and B. Ding. 2021. CGM: An Enhanced Mechanism for Streaming Data Collection with Local Differential Privacy. *Proc. of VLDB Endow.* 14, 11 (2021), 2258–2270.

[5] J. Bolot, N. Fawaz, S. Muthukrishnan, A. Nikolov, and N. Taft. 2013. Private decayed predicate sums on streams. In *Proc. ACM ICDT.* 284–295.

[6] T.-H. Hubert Chan, E. Shi, and D. Song. 2011. Private and Continual Release of Statistics. *ACM Trans. Inf. Syst. Secur.* 2010 (2011), 76.

[7] Y. Chen, A. Machanavajjhala, M. Hay, and G. Miklau. 2017. PeGaSus: Data-Adaptive Differentially Private Stream Processing. *Proc. ACM CCS* (2017), 1375–1388.

[8] G. Cormode, T. Kulkarni, and D. Srivastava. 2018. Marginal release under local differential privacy. In *Proc. ACM SIGMOD.* 131–146.

[9] G. Cormode, T. Kulkarni, and D. Srivastava. 2019. Answering range queries under local differential privacy. *Proc. of VLDB Endow.* 12, 10 (2019), 1126–1138.

[10] G. Cormode, S. Maddock, and C. Maple. 2021. Frequency estimation under local differential privacy. *Proc. of VLDB Endow.* 14, 11 (2021), 2046–2058.

[11] differential privacy team at Apple. 2017. Learning with privacy at scale. https://machinelearning.apple.com/research/learning-with-privacy-at-scale

[12] B. Ding, J. Kulkarni, and S. Yekhanin. 2017. Collecting telemetry data privately. In *Proc. NeurIPS.* 3574–3583.

[13] J. Duchi, M. Jordan, and M. Wainwright. 2013. Local privacy and statistical minimax rates. In *Proc. IEEE FOCS.* 429–438.

[14] J. Duchi, M. Jordan, and M. Wainwright. 2014. Privacy aware learning. *J. ACM* 61, 6 (2014), 1–57.

[15] J. Duchi, M. Wainwright, and M. Jordan. 2016. Minimax Optimal Procedures for Locally Private Estimation. *J. Am. Stat. Assoc* 113 (2016), 182 – 201.

[16] C. Dwork. 2006. Differential privacy. In *Proc. ICALP.* 1–12.

[17] C. Dwork. 2010. Differential Privacy in New Settings. In *Proc. ACM-SIAM SODA.* 174–183.

[18] C. Dwork, M. Naor, T. Pitassi, and G. N. Rothblum. 2010. Differential Privacy under Continual Observation. In *Proc. ACM STOC.* 715–724.

[19] C. Dwork and A. Roth. 2014. The Algorithmic Foundations of Differential Privacy. *Found. Trends Theor. Comput. Sci.* 9 (2014), 211–407.

[20] Ú. Erlingsson, V. Feldman, I. Mironov, A. Raghunathan, K. Talwar, and A. Thakurta. 2019. Amplification by shuffling: From local to central differential privacy via anonymity. In *Proc. ACM-SIAM SODA.* 2468–2479.

[21] Ú. Erlingsson, A. Korolova, and V. Pihur. 2014. RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response. In *Proc. ACM CCS.* 1054–1067.

[22] L. Fan and L. Xiong. 2012. Real-time aggregate monitoring with differential privacy. In *Proc. ACM CIKM.* 2169–2173.

[23] L. Fan and L. Xiong. 2014. An Adaptive Approach to Real-Time Aggregate Monitoring With Differential Privacy. *IEEE Trans. on Knowl. Data Eng.* 26 (2014), 2094–2106.

[24] L. Fan, L. Xiong, and V. S. Sunderam. 2013. Differentially Private Multi-dimensional Time Series Release for Traffic Monitoring. In *Proc. DBSec.* 33–48.

[25] N. Johnson, J. Near, and D. Song. 2018. Towards practical differential privacy for SQL queries. *Proc. VLDB Endow.* 11, 5 (2018), 526–539.

[26] M. Joseph, A. Roth, J. Ullman, and B. Waggoner. 2018. Local Differential Privacy for Evolving Data. *Proc. NeurIPS* 31 (2018), 2375–2384.

[27] P. Kairouz, S. Oh, and P. Viswanath. 2016. Extremal mechanisms for local differential privacy. *J. Mach. Learn. Res.* 17, 1 (2016), 492–542.

[28] S. Kasiviswanathan, H. Lee, N, S. Raskhodnikova, and A. Smith. 2011. What can we learn privately? *SIAM J. Computing* 40, 3 (2011), 793–826.

[29] G. Kellaris, S. Papadopoulos, X. Xiao, and D. Papadias. 2014. Differentially private event sequences over infinite streams. *Proc. VLDB Endow.* 7 (2014), 1155–1166.

[30] H. Li, L. Xiong, X. Jiang, and J. Liu. 2015. Differentially Private Histogram Publication for Dynamic Datasets: An Adaptive Sampling Approach. In *Proc. ACM CIKM.* 1001–1010.

[31] F. McSherry. 2009. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *Proc. ACM SIGMOD.* 19–30.

[32] T. Murakami and Y. Kawamoto. 2019. Utility-optimized local differential privacy mechanisms for distribution estimation. In *Proc. USENIX Security.* 1877–1894.

[33] T. Nguyên, X. Xiao, Y. Yang, S. Hui, H. Shin, and J. Shin. 2016. Collecting and analyzing data from smart device users with local differential privacy. *arXiv preprint arXiv:1606.05053* (2016).

[34] V. Perrier, H. Asghar, and Dali Kaafar. 2019. Private Continual Release of Real-Valued Data Streams. In *Proc. NDSS.*

[35] Z. Qin, Y. Yang, T. Yu, I. Khalil, X. Xiao, and K. Ren. 2016. Heavy Hitter Estimation over Set-Valued Data with Local Differential Privacy. In *Proc. ACM CCS.* 192–203.

[36] Z. Qin, T. Yu, Y. Yang, I. Khalil, X. Xiao, and K. Ren. 2017. Generating synthetic decentralized social graphs with local differential privacy. In *Proc. ACM CCS.* 425–438.

[37] X. Ren, L. Shi, W. Yu, S. Yang, C. Zhao, and Z. Xu. 2022. LDP-IDS: Local Differential Privacy for Infinite Data Streams. *Technical Report, available online at https://anonymous.4open.science/r/Paper_LDP-IDS/LDP-IDS.pdf* (2022).

[38] X. Ren, C.-M. Yu, W. Yu, S. Yang, X. Yang, J. McCann, and S. Philip. 2018. LoPub: High-Dimensional Crowdsourced Data Publication with Local Differential Privacy. *IEEE Trans. Inf. Forensics Security* 13, 9 (2018), 2151–2166.

[39] X. Ren, C.-M. Yu, W. Yu, X. Yang, J. Zhao, and S. Yang. 2020. DPCrowd: privacy-preserving and communication-efficient decentralized statistical estimation for real-time crowdsourced data. *IEEE Internet Things J.* 8, 4 (2020), 2775–2791.

[40] J. Tang, A. Korolova, X. Bai, X. Wang, and X. Wang. 2017. Privacy loss in apple's implementation of differential privacy on macos 10.12. *arXiv preprint arXiv:1709.02753* (2017).

[41] N. Wang, X. Xiao, Y. Yang, T.-D. Hoang, H. Shin, J. Shin, and G. Yu. 2018. PrivTrie: Effective frequent term discovery under local differential privacy. In *Proc. IEEE ICDE.* 821–832.

[42] N. Wang, X. Xiao, Y. Yang, J. Zhao, S. Hui, H. Shin, J. Shin, and G. Yu. 2019. Collecting and Analyzing Multidimensional Data with Local Differential Privacy. In *Proc. IEEE ICDE.* 638–649.

[43] Q. Wang, Y. Zhang, X. Lu, Z. Wang, Z. Qin, and K. Ren. 2016. RescueDP: Real-time spatio-temporal crowd-sourced data publishing with differential privacy. *Proc. IEEE INFOCOM* (2016), 1–9.

[44] T. Wang, J. Blocki, N. Li, and S. Jha. 2017. Locally Differentially Private Protocols for Frequency Estimation. In *Proc. USENIX Security.* 729–745.

[45] T. Wang, J.-Q. Chen, Z. Zhang, D. Su, Y. Cheng, Z. Li, N. Li, and S. Jha. 2021. Continuous Release of Data Streams under both Centralized and Local Differential Privacy. In *Proc. ACM CCS.* 1237–1253.

[46] T. Wang, B. Ding, J. Zhou, C. Hong, Z. Huang, N. Li, and S. Jha. 2019. Answering multi-dimensional analytical queries under local differential privacy. In *Proc. ACM SIGMOD.* 159–176.

[47] T. Wang, N. Li, and S. Jha. 2019. Locally differentially private heavy hitter identification. *IEEE Trans. Dependable Secure Comput.* (2019).

[48] Z. Wang, W. Liu, X. Pang, J. Ren, Z. Liu, and Y. Chen. 2020. Towards Pattern-aware Privacy-preserving Real-time Data Collection. In *Proc. IEEE INFOCOM.* 109–118.

[49] Z. Wang, X. Pang, Y. Chen, H. Shao, Q. Wang, L. Wu, H. Chen, and H. Qi. 2018. Privacy-preserving crowd-sourced statistical data publishing with an untrusted server. *IEEE Trans. Mobile Computing* 18, 6 (2018), 1356–1367.

[50] Q. Ye, H. Hu, X. Meng, and H. Zheng. 2019. PrivKV: Key-value data collection with local differential privacy. In *Proc. IEEE SP.* 317–331.

[51] Z. Zhang, T. Wang, N. Li, S. He, and J. Chen. 2018. CALM: Consistent Adaptive Local Marginal for Marginal Release under Local Differential Privacy. In *Proc. ACM CCS.* 212–229.