Money Cannot Buy Everything: Trading Mobile Data with Controllable Privacy Loss

Shuyuan Zheng, Yang Cao✉, Masatoshi Yoshikawa
Department of Social Informatics
Kyoto University
Kyoto, Japan
{caryzheng@db., yang@, yoshikawa@}i.kyoto-u.ac.jp

***Abstract*—As personal data has been the new oil of the digital era, there is a growing trend perceiving personal data as a commodity. Existing studies have built theories on how to map the privacy loss to an arbitrage-free price. They assumed that a data buyer could purchase arbitrarily accurate results as long as she could compensate data owners for their privacy loss. However, it may not be a viable business model under strict privacy regulations, such as GDPR and CCPA, and data owners' emerging privacy concerns. In this paper, we study how to empower data owners with the control of privacy loss when continuously trading their personal mobile data. Concretely, we propose a framework for trading infinite streaming mobile data which enables each data owner to bound her privacy loss in a w-length sliding window. Introducing such upper bounds of privacy loss makes the existing trading frameworks invalid and raises new technical challenges in terms of budget allocation and arbitrage-free pricing. To address these problems, we propose a modularized trading framework with instances that allows data owners to personalize their privacy loss while the price is still arbitrage-free. Finally, we conduct experiments to verify the effectiveness of the proposed trading protocols.**

***Keywords*-personal data trading, personalized differential privacy, budget allocation, arbitrage-free pricing**

I. Introduction

Personal data is widely exploited by companies to improve products and services. Since personal data may release sensitive information that can be abused for crimes, data owners also deserve appropriate compensations due to their potential *privacy loss*. In fact, there is a growing trend towards *personal data trading* as a new business model. This model allows data owners to trade off privacy loss for money, which benefits personal data circulation and utilization.

Several studies [1]–[4] in the literature investigated privacy-preserving query-based data trading, as shown in Figure 1. There are three parties in the data marketplace: *data owners*, *data buyers*, and a *broker*. Data owners contribute their personal data and get monetary compensations in return. Data buyers request queries over the data and pay for perturbed query answers where some random noises are injected. The broker acts as a trustworthy intermediary between data owners and data buyers, in charge of computing perturbed query answers, setting *query prices*, and compensating data owners. A major challenge in this line of works is how to price queries. A seminal work of Li et al. [2] assigned prices to queries according to their utility. They designed pricing functions

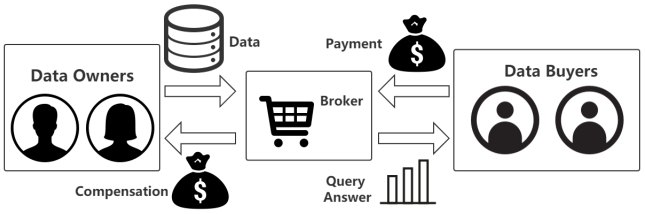

Fig. 1. Query-based Data Trading

by making the connection between the privacy loss and the utility; then they formulated an important property of pricing functions: *arbitrage-freeness*, which means the consistency of a set of priced queries. Intuitively, a data buyer should not obtain the answer to a query by deriving this answer from a less expensive set of queries.

However, the previously proposed data marketplaces did not give data owners the power to control their privacy loss in continuous data trading. Concretely, there are several desiderata that have not been well studied. First, data owners should be able to specify an *upper bound* of privacy loss given the high sensitivity of mobile data such as human trajectories. In the traditional data marketplace shown in Figure 1, a data buyer even can purchase a query answer with arbitrary accuracy as long as she can pay for it, which theoretically means she can purchase the raw data. However, our recent survey [5] reveals that most of data owners value privacy or other criteria more important than the financial compensation when monetizing their data, which supports our argument that money cannot buy everything. Second, in the data trading, each data owner should be able to choose a *personalized* privacy loss bound she likes; however, the existing studies [1]–[3] enforce a uniform privacy loss for all data owners. Third, existing studies mainly focus on trading static personal data; however, large amounts of personal data are generated *continuously*.

In this paper, we study how to empower data owners with the control of privacy loss when continuously trading their personal mobile data. There are three challenges in solving the problems mentioned above. First, bounding the privacy loss indicates restricting the maximum supply of the utility in the data marketplace. There is no framework in the existing studies for trading data under utility constraints. This leads to a new and challenging problem: how to trade query answers when their utilities are bounded.

Second, personalized privacy losses make it more difficult to design an arbitrage-free pricing function. As the price depends on the query answer's utility while the cost of the answer (i.e., the compensations to data owners) is decided by privacy losses, we design a cost-covering pricing function by establishing a one-to-one connection between the utility and privacy losses. Under uniform privacy losses, higher privacy losses mean better utility; thus, such a one-to-one connection always exists and can even naturally fit the requirements of arbitrage-free pricing. However, under personalized privacy losses, the connection might be one-to-many. It also might be hard to express such a connection in analytic form, study the properties of its corresponding pricing function, and improve it to guarantee arbitrage-free pricing. Although we can leverage *personalized differential privacy* (PDP) [6] to realize personalized privacy losses, no existing work has studied arbitrage-free pricing under PDP techniques, which is a non-trivial challenge.

Third, trading *infinite streaming* mobile data brings more complexity to arbitrage-free pricing. As the broker should protect human trajectories, the task of privacy budget allocation arises, concerning not only the privacy losses at a single time-point but also on the timeline. Although w-event privacy [7] and some budget allocation techniques have been proposed to protect sensitive information revealed from multiple time-points, the problem of how to design an arbitrage-free pricing function with constraints from both w-event privacy and PDP is hugely complicated.

Contribution

In this paper, for the first time, we study how to design a personal data trading framework that empowers data owners with the control of privacy loss. Our main contributions are summarized as follows. First, we design a mobile data trading framework with three main steps in the protocol: *Offer*, *Quote*, and *Delivery*, to match supply (i.e., bounded privacy losses from data owners) and demand (i.e., utility requirements from data buyers) in a data marketplace. The major challenge is how to ensure arbitrage-free pricing under personalized privacy losses. Essentially, it requires the broker to derive the privacy losses given a required utility (i.e., variance), which is non-trivial if the differentially private mechanism is complex. We present a general method to design a pricing function based on the inverse function of the utility function of query answers; then, we propose a method to adjust the given privacy budgets from data owners to make the pricing function arbitrage-free.

Second, we develop a principled way to instantiate the protocol of our trading framework. Specifically, we propose a general theorem (Theorem 3.1) presenting that if the utility function of query answers satisfies certain properties, it will result in an arbitrage-free pricing function. This theorem guides the design of trading protocols for our framework centering around arbitrage-free pricing. Based on this theorem, we transform the problem of designing an arbitrage-free pricing function into finding a utility function satisfying such properties.

Third, we propose two arbitrage-free trading protocols by instantiating three key modules in our framework. We transform the problem of designing an arbitrage-free pricing function into an optimization problem. Then, for our advanced protocol, we design a privacy budget allocation algorithm with a pre-processing algorithm solving this optimization problem, in order to guarantee arbitrage-free pricing. Our experimental results verify the effectiveness of our advanced protocol.

The rest of this paper is organized as follows. Section 2 introduces some basic settings. Section 3 gives a full view of our trading framework. Section 4 discusses the technical details of our arbitrage-free trading protocols. Section 5 presents the experimental results. Section 6 introduces related works, and finally Section 7 draws a conclusion.

II. Problem Formulation

In this section, we introduce the basic settings of our mobile data marketplace and outline our design goals in the end.

A. Preliminaries

Trajectory data stream.: At every time-point τ, data owners contribute a trajectory database D^τ with n rows, where each row (a trajectory data point) corresponds to a unique data owner u_i and $D^\tau[i] \in \{1, ..., d\}$ represents the ID of u_i's location at time-point τ. A trajectory data stream $S = (D^1, D^2, ...)$ is an infinite sequence of trajectory databases. A stream prefix $S^t = (D^1, ..., D^t)$ is the prefix of S, where variable $t \in [\tau]$ is an index of time-point. We note that τ denotes the most recent time-point and D^τ denotes the most recently released trajectory database.

Histogram query: In our settings, a data buyer can request a histogram query Q^τ over the trajectory database D^τ. The trajectory database D^τ can be transformed into a raw query answer $q^\tau = (q_1^\tau, ..., q_d^\tau)$ where each element q_l^τ represents the total number of data owners who were in the location of ID $l \in [d]$ at time-point τ.

Personalized differential privacy (PDP): To preserve data owners' privacy, the broker cannot return raw query answers to data buyers and should perturb them by some perturbation mechanism $\mathcal{M}$ achieving a formal privacy standard. Since each data owner is allowed to set a personalized upper bound of her privacy loss, we deploy personalized differential privacy (a variant of differential privacy [8]), where each data owner u_i corresponds to a personalized privacy protection level, i.e., a privacy loss ϵ_i. [6] proposed the Sample mechanism to achieve PDP, and showed that the Laplace mechanism [8] can also be used to achieve it but results in uniform privacy losses.

Definition 2.1 (Personalized Differential Privacy [6]): Given a privacy specification $\Phi = [\epsilon_1, ..., \epsilon_n]$, a perturbation mechanism $\mathcal{M}$ satisfies Φ-personalized differential privacy, if for any pair of neighboring databases D, D' with $D \overset{u_i}{\sim} D'$, and for any possible output $o \in \mathcal{R}^d$, we have: $Pr[\mathcal{M}(D, \Phi) = o] \leq e^{\epsilon_i} Pr[\mathcal{M}(D', \Phi) = o]$.

Two databases D, D' are neighboring if D' can be derived from D by replacing one row with another, denoted as $D \sim D'$. We also write $D \overset{u_i}{\sim} D'$ to denote that D and D' are neighboring and differ on the value of $D[i]$.

Utility metric: To evaluate the utility of a perturbed query answer $\tilde{q}^\tau = (\tilde{q}_1^\tau, ..., \tilde{q}_d^\tau)$, we use histogram variance v as a metric, which is no lower than the maximum variance among all the elements of $\tilde{q}^\tau$, i.e., $v \geq \max_l Var(\tilde{q}_l^\tau)$. Variance has been used as a utility metric in several privacy-preserving data marketplaces [2]–[4], as it reflects the expected dispersion of a random variable (i.e., a perturbed query answer in our cases). We use histogram variance because our query answers are vectors rather than scalars and it provides a worst-case guarantee for all elements of a vector.

B. Market Setup

There are three parties of participants in our marketplace: data owners, data buyers, and a broker (who acts as a trustworthy intermediary between data owners and data buyers). Each party has its own interests and goals.

Data owners are the source of trajectory data points stored in the marketplace. Each data owner u_i should specify her own privacy loss bound $\hat{\epsilon}_i$ (i.e., the maximum privacy loss she can tolerate for every w_i successive time-points), and also the sliding-window size w_i.

Data buyers can request a histogram query Q^τ (where τ is the most recent time-point) over the database D^τ with a specific utility (i.e., a histogram variance v) until $D^{\tau+1}$ is released, and get a perturbed query answer $\tilde{q}^\tau$ such that $v \geq \max_l Var(\tilde{q}_l^\tau)$. For simplicity, we assume there is only one query Q^τ for every time-point τ; however, our techniques support multiple queries at a single time-point because we can easily combine privacy losses together according to the Composition theorem of PDP [6].

The broker and data owners: For preserving data owners' privacy, the broker is in charge of perturbing raw query answers by a perturbation mechanism $\mathcal{M}$ which achieves personalized differential privacy (PDP, see Definition 2.1). Since PDP only protects a single trajectory database rather than the entire stream, we present personalized $\mathcal{W}$-event privacy (PWP, see Definition 2.2) based on PDP [6] and w-event privacy [7] to ensure that, for each data owner u_i, the total privacy loss for every w_i successive time-points is bounded by $\hat{\epsilon}_i$.

Definition 2.2 (Φ-Personalized $\mathcal{W}$-event Privacy (PWP)): Given a privacy specification $\Phi = [\epsilon_1, ..., \epsilon_n]$, a window specification $\mathcal{W} = [w_1, ..., w_n]$, a mechanism $\mathcal{M}$ taking a stream prefix as input satisfies Φ-personalized $\mathcal{W}$-event privacy, if for any time-point τ, for any pair of w_i-neighboring stream prefixes $S^\tau, S^{\tau\prime}$, with either $S^\tau[t] = S^{\tau\prime}[t]$ or $S^\tau[t] \overset{u_i}{\sim} S^{\tau\prime}[t]$ for every $t \in [\tau]$, and for any possible output o, we have $Pr[\mathcal{M}(S^\tau, \Phi) = o] \leq e^{\epsilon_i} Pr[\mathcal{M}(S^{\tau\prime}, \Phi) = o]$.

Two stream prefixes $S^\tau, S^{\tau\prime}$ are w_i-neighboring [7] if the follow conditions are satisfied:

1) there are at most w_i pairs of neighboring databases $S^\tau[t]$, $S^{\tau\prime}[t]$, $t \in [\tau]$;
2) if two databases $S^\tau[t]$, $S^{\tau\prime}[t]$ are not neighboring, then $S^\tau[t] = S^{\tau\prime}[t]$;
3) all the pairs of neighboring databases $S^\tau[t]$, $S^{\tau\prime}[t]$ are in a w_i-length window.

Contract between each data owner and the broker: The broker should compensate data owners according to their privacy losses. Before data trading, each data owner u_i should make a contract with the broker that the latter is obligated to compensate the former by a compensation function μ_i. Let ϵ_i^τ be data owner u_i's privacy loss caused by the perturbed query answer $\tilde{q}^\tau$. Each data owner's compensation $\mu_i(\epsilon_i^\tau)$ depends on her privacy loss ϵ_i^τ, i.e., $\mu_i(\epsilon_i^\tau) = c_i \cdot \epsilon_i^\tau$ where $c_i > 0$ is a constant compensation rate stipulated in the contract.

The broker and data buyers: As a profit-making intermediary, the broker sells perturbed query answers to make a profit. Because data buyers care more about the utility of query answers, we set that the price of a perturbed query answer directly depends on its histogram variance v, rather than on the privacy losses it caused. The broker should design a pricing function $\pi = \Pi(v)$ which is: (1) *cost-covering*, which means a query price should cover the total compensations to data owners, and (2) *arbitrage-free*, which means a data buyer cannot obtain a perturbed query answer with a specific histogram variance v more cheaply by deriving it from a less expensive set of perturbed query answers.

Definition 2.3 (Cost-covering): A pricing function $\pi = \Pi(v)$ $(v > 0)$ is cost-covering, if it satisfies the following condition: for any perturbed query answer $\tilde{q}$ with a histogram variance v which caused a privacy loss ϵ_i for each data owner u_i, we have $\pi = \Pi(v) = \sum_i \mu_i(\epsilon_i)$.

We note that we make the price equal to the total compensations for simplicity, but our techniques also can support those pricing functions that additionally cover a profit.

Definition 2.4 (Arbitrage-freeness): A pricing function $\pi = \Pi(v)$ is arbitrage-free, if it satisfies the following condition: for every multiset $V = \{v_1, ..., v_m\}$ where $m \in Z^+$, if there exists $a_1, ..., a_m$ such that $\sum_{j=1}^m a_j = 1$ and $\sum_{j=1}^m a_j^2 v_j \leq v$, we have $\Pi(v) \leq \sum_{j=1}^m \Pi(v_j)$.

Goals. We summarize our design goals for each party:

- *Data owners:* We try to consume their privacy losses within their privacy loss bounds as much as possible so that they can gain higher compensations.
- *Data buyers:* As data owners' privacy losses are bounded, the histogram variance of a query answer is also bounded. Thus, a data buyer may fail to purchase a query answer with a very low histogram variance. One of our design goals is to lower the lower bound of histogram variance to make more queries answerable. Besides, we try to improve the cost-performance, i.e., to provide a lower query price for the same histogram variance v.
- *The broker:* The pricing function should be arbitrage-free and cost-covering.

III. Mobile Data Trading Framework

In this section, first we give a short view of our mobile data trading framework. Then, we discuss more details about arbitrage-free pricing and three key modules in our framework. In a nutshell, we provide a framework to compute arbitrage-free trading protocols. Some important notations are summarized in Table I.

TABLE I
SUMMARY OF NOTATIONS

Notation	Description
u_i	data owner i
n	total number of users
t	index of time-point
τ	the most recent time-point
D^τ	trajectory database collected at τ
Q^τ	histogram query over D^τ
$\tilde{q}^\tau$	perturbed query answer to Q^τ
v^τ	histogram variance of $\tilde{q}^\tau$
$\check{v}^\tau$	lower bound of v^τ
$\hat{\epsilon}_i$	u_i's privacy loss bound
$\hat{\epsilon}_i^\tau$	u_i's raw privacy budget for Q^τ
$\bar{\epsilon}_i^\tau$	u_i's adjusted privacy budget for Q^τ
ϵ_i^τ	u_i's privacy loss for Q^τ
$\mu_i(\cdot)$	u_i's compensation function
$\mathcal{M}$	perturbation mechanism
$v = U_\mathcal{M}([\epsilon_1, ..., \epsilon_n])$	utility function under $\mathcal{M}$
$\pi = \Pi_\mathcal{M}(v)$	pricing function under $\mathcal{M}$

A. Framework Overview

A transaction in our data marketplace works as depicted in Figure 2. After the release of D^τ, the broker should run two budget allocation modules briefly introduced as follows:

- *BudgetAlloc* (budget allocation): It allocates raw privacy budgets $[\hat{\epsilon}_1^\tau, ..., \hat{\epsilon}_n^\tau]$, with the goals of not only (1) making full use of data owners' privacy losses within their privacy loss bounds but also (2) making more queries answerable. We note that a privacy budget is the budget of privacy loss for a single time-point while a privacy loss bound works for multiple time-points.
- *BudgetAdjusting* (budget adjusting): It adjusts raw privacy budgets to adjusted privacy budgets $[\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau]$ to guarantee arbitrage-free pricing. We will discuss why we need to adjust raw privacy budgets in Section III-B. We note that each data owner's adjusted privacy budget $\bar{\epsilon}_i^\tau$ should be no higher than her raw privacy budget $\hat{\epsilon}_i^\tau$.

Once the adjusted privacy budgets $[\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau]$ are computed, the broker starts to monitor query requests over D^τ and trade perturbed query answers in the following steps.

Offer: In Step 1, a data buyer requests a query Q^τ over database D^τ. Then in Step 2, the broker returns the maximum utility to the data buyer, i.e., the lower bound $\check{v}^\tau$ of histogram variance v^τ. Here we define the utility function of query answers perturbed by $\mathcal{M}$ as $v = U_\mathcal{M}([\epsilon_1, ..., \epsilon_n])$ which takes as input a vector of privacy losses (or budgets) and outputs the histogram variance v. The value of $\check{v}^\tau$ depends on the adjusted privacy budgets, i.e., $\check{v}^\tau = U_\mathcal{M}([\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau])$.

Quote: In Step 3, given the lower bound $\check{v}^\tau$, the data buyer should choose a histogram variance $v^\tau \geq \check{v}^\tau$. Then in Step 4, the broker quotes a price π^τ for a perturbed query answer $\tilde{q}^\tau = Q^\tau(D^\tau, v^\tau)$ using an arbitrage-free and cost-covering pricing function $\pi = \Pi_\mathcal{M}(v)$.

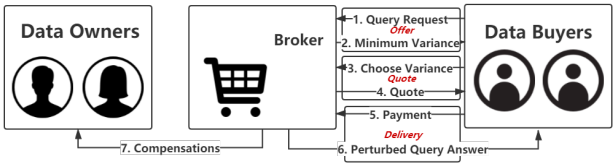

Fig. 2. Transaction Flow in our Trading Framework

Delivery: In Step 5, the data buyer pays the price π^τ. Then in Step 6, the broker calculates data owners' privacy losses $[\epsilon_1^\tau, ..., \epsilon_n^\tau]$ for v^τ, returns a perturbed query answer $\tilde{q}^\tau = \mathcal{M}(D^\tau, [\epsilon_1^\tau, ..., \epsilon_n^\tau])$ to the data buyer, and compensates each data owner according to her own privacy loss. Each data owner u_i's privacy loss ϵ_i^τ should be no higher than her adjusted privacy budget $\bar{\epsilon}_i^\tau$ because the latter is the budget of the former. We note that a perturbation mechanism $\mathcal{M}$ is an instance of module *PerturbMech* which can satisfy personalized differential privacy. Thus we have three key modules: *PerturbMech*, *BudgetAlloc*, and *BudgetAdjusting*.

B. Arbitrage-free Pricing under Utility Constraints

In Section III-B, we discuss in detail the design of arbitrage-free pricing in our framework. Since pricing is the most important issue in a marketplace, it guides the design of our key modules. Before the discussion of how we can guarantee arbitrage-free pricing, we introduce how we design a pricing function in our framework.

Inverse function: Remind that in Step 6, given a histogram variance v^τ, the broker should calculate the corresponding privacy losses $[\epsilon_1^\tau, ..., \epsilon_n^\tau]$, and run $\mathcal{M}(D^\tau, [\epsilon_1^\tau, ..., \epsilon_n^\tau])$ to compute a perturbed query answer. The question arises how to calculate the corresponding privacy losses given a histogram variance, which means we need a mapping from the set of histogram variance to the set of privacy losses. Because the utility function $v = U_\mathcal{M}([\epsilon_1, ..., \epsilon_n])$ is a mapping from the set of privacy losses to the set of histogram variance, the inverse function of the utility function, i.e., ${U_\mathcal{M}}^{-1}$, is the mapping we need. Thus, we guarantee the existence of ${U_\mathcal{M}}^{-1}$ so that we can derive the corresponding privacy losses.

To design the pricing function by the inverse function: Given the inverse function $[\epsilon_1, ..., \epsilon_n] = {U_\mathcal{M}}^{-1}(v)$, the design of the pricing function has an anchor point, i.e., the compensations to data owners. Remind that a query price should cover the total compensations to data owners, i.e., $\pi = \Pi_\mathcal{M}(v) \geq \sum_i c_i \cdot \epsilon_i$. The direct way of designing a pricing function is to make the query price equal to the sum of compensations. Thus, we can derive a cost-covering pricing function as $\pi = \Pi_\mathcal{M}(v) = \sum_i c_i \cdot \epsilon_i = [c_1, ..., c_n] \cdot [\epsilon_1, ..., \epsilon_n] = [c_1, ..., c_n] \cdot {U_\mathcal{M}}^{-1}(v)$. We note that the profit-making broker can simply modify this pricing function to derive another one that covers a profit, e.g., $\pi = \Pi_\mathcal{M}(v) = (1 + r)[c_1, ..., c_n] \cdot {U_\mathcal{M}}^{-1}(v)$ where $r > 0$ is a flat profit rate; we employ $\pi = \Pi_\mathcal{M}(v) = [c_1, ..., c_n] \cdot {U_\mathcal{M}}^{-1}(v)$ for simplicity but our techniques support the former case.

To make the inverse function existent: The inverse function under a simple perturbation mechanism, such as the Laplace mechanism [8], can be easy to derive. However, it may not

exist under a sophisticated mechanism such as the Sample mechanism [6]. To guarantee the existence of ${U_{\mathcal{M}}}^{-1}$, first, we introduce the notion ρ-Pattern to constrain the domain of the utility function's input. Intuitively, the inverse function does not exist because an output of the utility function corresponds to multiple inputs. Hence, we can constrain the domain to rule out those redundant inputs, so that there is a bijection between the sets of histogram variance and privacy losses.

Definition 3.1 (ρ-Pattern): A pattern is a vector $\rho = [\rho_1, ..., \rho_n]$ where $0 \leq \rho_i \leq 1$ for all i and $\exists j \in [n]$ such that $\rho_j = 1$. If a vector $[\epsilon_1, ..., \epsilon_n]$ is equal to $\rho \cdot \max_i \epsilon_i$, then the vector is in ρ-pattern. We also use ϵ_{max} to denote $\max_i \epsilon_i$.

With the introduction of ρ-Pattern, we can use a fixed pattern ρ to constrain the domain of the utility function so that every input $[\epsilon_1, ..., \epsilon_n]$ is in ρ-pattern, i.e., $[\epsilon_1, ..., \epsilon_n] = \rho \cdot \epsilon_{max}$. That also means, for every time-point τ, privacy budgets or privacy losses should be in a fixed ρ-pattern, which makes $U_{\mathcal{M}}([\epsilon_1, ..., \epsilon_n]) = U_{\mathcal{M}}(\rho \cdot \epsilon_{max})$; otherwise, privacy budgets should be adjusted to fit in ρ-pattern. For example, at time-point 1, the broker allocates adjusted privacy budgets $[\bar{\epsilon}_1^1, \bar{\epsilon}_2^1, \bar{\epsilon}_3^1] = [4, 2, 2]$ and the data buyer consumes privacy losses $[\epsilon_1^1, \epsilon_2^1, \epsilon_3^1] = [2, 1, 1]$; at time-point 2, $[\bar{\epsilon}_1^2, \bar{\epsilon}_2^2, \bar{\epsilon}_3^2] = [6, 3, 3]$ and $[\epsilon_1^2, \epsilon_2^2, \epsilon_3^2] = [4, 2, 2]$; all those privacy budgets and losses are in $[1, 0.5, 0.5]$-pattern. Then we guarantee the existence of ${U_{\mathcal{M}}}^{-1}$ by finding a ρ such that $U_{\mathcal{M}}(\rho \cdot \epsilon_{max})$ decreases as ϵ_{max} increases, which is also implied by the first property in Theorem 3.1.

To guarantee arbitrage-free pricing by ρ-Pattern: Now we can derive a pricing function as follows:

$$\pi = \Pi_{\mathcal{M}}(v) = [c_1, ..., c_n] \cdot {U_{\mathcal{M}}}^{-1}(v) = [c_1, ..., c_n] \cdot \rho \cdot \epsilon_{max}.$$

However, under some perturbation mechanisms $\mathcal{M}$, the utility function can be so complex that the inverse function even cannot be expressed in analytic form, which makes it difficult to analyze the mathematical properties of the pricing function $\Pi_{\mathcal{M}}$ and then guarantee its arbitrage-freeness. Hence, we consider whether we can inversely study the properties of the utility function $U_{\mathcal{M}}$ to guarantee arbitrage-free pricing, and propose Theorem 3.1 illustrating that: if $U_{\mathcal{M}}$ satisfies certain properties, then $\Pi_{\mathcal{M}}$ will be arbitrage-free.

Theorem 3.1: Given a pattern ρ, the pricing function $\pi = \Pi_{\mathcal{M}}(v)$ is arbitrage-free, if the utility function $v = U_{\mathcal{M}}(\rho \cdot \epsilon_{max})$ satisfies the following properties:

1) $\forall \epsilon_{max} > 0$, ${U_{\mathcal{M}}}'(\rho \cdot \epsilon_{max}) < 0$;
2) $\forall \epsilon_{max} > 0$, $U_{\mathcal{M}}(\rho \cdot \epsilon_{max}) \cdot {U_{\mathcal{M}}}''(\rho \cdot \epsilon_{max}) \leq 2[{U_{\mathcal{M}}}'(\rho \cdot \epsilon_{max})]^2$;
3) $\lim_{\epsilon_{max} \to 0^+} U_{\mathcal{M}}(\rho \cdot \epsilon_{max}) = +\infty$.

(${U_{\mathcal{M}}}'$ and ${U_{\mathcal{M}}}''$ are the first and second order derivatives of ϵ_{max}, respectively.)

Proof: First, we construct a function $v = U_{\mathcal{M},\rho}(\epsilon_{max})$ such that $U_{\mathcal{M},\rho}(\epsilon_{max}) = U_{\mathcal{M}}(\rho \cdot \epsilon_{max})$. We drop the subscripts $\mathcal{M}$ and max from $U_{\mathcal{M}}$, $U_{\mathcal{M},\rho}$, $\Pi_{\mathcal{M}}$ and ϵ_{max} to simplify notation. Then, we construct the following function:

$$h(x) = \begin{cases} U_\rho^{-1}(\frac{1}{x}), & x > 0 \\ 0, & x = 0. \end{cases}$$

Because of the first property in the theorem, we can derive that $\forall v > 0, {U_\rho^{-1}}'(v) < 0$. Then, because of the second property in the theorem, we have $h''(x) \leq 0$ for all $x > 0$:

$$\begin{aligned}
&\forall \epsilon > 0, U(\rho \cdot \epsilon) \cdot U''(\rho \cdot \epsilon) - 2 \cdot [U'(\rho \cdot \epsilon)]^2 \leq 0 \\
\Rightarrow &\forall \epsilon > 0, U_\rho(\epsilon) \cdot U_\rho''(\epsilon) - 2 \cdot [U_\rho'(\epsilon)]^2 \leq 0 \\
\Rightarrow &\forall v > 0, v \cdot [-\frac{{U_\rho^{-1}}''(v)}{({U_\rho^{-1}}'(v))^3}] - 2 \cdot (\frac{1}{{U_\rho^{-1}}'(v)})^2 \leq 0 \\
\Rightarrow &\forall v > 0, v \cdot {U_\rho^{-1}}''(v) + 2 \cdot {U_\rho^{-1}}'(v) \leq 0 \\
\Rightarrow &\forall x > 0, \frac{1}{x} \cdot {U_\rho^{-1}}''(\frac{1}{x}) + 2 \cdot {U_\rho^{-1}}'(\frac{1}{x}) \leq 0 \\
\Rightarrow &\forall x > 0, h''(x) = {U_\rho^{-1}}''(\frac{1}{x}) \cdot [(\frac{1}{x})']^2 + {U_\rho^{-1}}'(\frac{1}{x}) \cdot (\frac{1}{x})'' \leq 0.
\end{aligned}$$

Then, because of the third property in the theorem, $h(x)$ is right-continuous for $x = 0$:

$$\begin{aligned}
&\lim_{\epsilon -> 0^+} U(\rho \cdot \epsilon) = +\infty \\
\Rightarrow &\lim_{x -> 0^+} h(x) = \lim_{x -> 0^+} U_\rho^{-1}(\frac{1}{x}) = \lim_{v -> +\infty} U_\rho^{-1}(v) = 0 \\
\Rightarrow &\lim_{x -> 0^+} h(x) = h(0).
\end{aligned}$$

Then, according to Lagrange Mean Value Theorem, it can be implied that $h(x)$ is a sub-additive function:

$$\begin{aligned}
&\forall x_1, x_2 \in [0, +\infty), x_1 \leq x_2, \\
&h(x_1) + h(x_2) - h(x_1 + x_2) \\
=&[h(x_1) - h(0)] - [h(x_1 + x_2) - h(x_2)] \\
=&x_1 \cdot h'(\xi_1) - x_1 \cdot h'(\xi_2) = x_1 \cdot (\xi_1 - \xi_2) \cdot h''(\xi_3) \geq 0.
\end{aligned}$$

where $\xi_1 \in (0, x_1)$, $\xi_2 \in (x_2, x_1 + x_2)$, and $\xi_3 \in (\xi_1, \xi_2)$. Then, for any multiset $v_1, ..., v_m$ and any $a_1, ..., a_m$ such that $\sum_{j=1}^m a_j = 1$ and $\sum_{j=1}^m a_j^2 v_j \leq v$, we have:

$$\begin{aligned}
&U_\rho^{-1}(v) \leq U_\rho^{-1}(\sum_{j=1}^m a_j^2 v_j) = U_\rho^{-1}(\frac{\sum_{j=1}^m a_j^2 v_j}{(\sum_{j=1}^m a_j)^2}) \\
&\leq U_\rho^{-1}(\frac{\sum_{j=1}^m a_j^2 v_j}{(\sum_{j=1}^m a_j^2 v_j) \cdot (\sum_{j=1}^m \frac{1}{v_j})}) = U_\rho^{-1}(\frac{1}{\sum_{j=1}^m \frac{1}{v_j}}) \\
&= h(\sum_{j=1}^m \frac{1}{v_j}) \leq \sum_{j=1}^m h(\frac{1}{v_j}) = \sum_{j=1}^m U_\rho^{-1}(v_j).
\end{aligned}$$

Finally, we prove that $\pi = \Pi(v)$ is arbitrage-free:

$$\begin{aligned}
&\Pi(v) = [c_1, ..., c_n] \cdot U^{-1}(v) = [c_1, ..., c_n] \cdot \rho \cdot U_\rho^{-1}(v) \\
&\leq [c_1, ..., c_n] \cdot \rho \cdot \sum_{j=1}^m U_\rho^{-1}(v_j) \\
&= \sum_{j=1}^m [c_1, ..., c_n] \cdot U^{-1}(v_j) = \sum_{j=1}^m \Pi(v_j).
\end{aligned}$$

■

Hence, we can transform the problem of designing an arbitrage-free pricing function into finding a utility function satisfying such properties. Intuitively, the first property makes the histogram variance v decrease as ϵ_{max} increases, so that $\Pi_{\mathcal{M}}$ is a monotonically decreasing function. The second ensures a slow decreasing speed of $\Pi_{\mathcal{M}}$, so that purchasing a query answer with a lower v is more worthwhile than a higher

v. The third means that zero privacy losses correspond to an infinitely high histogram variance, which implies that no useful query answer can be traded for free. We note that, the selection of ρ is essentially the key to arbitrage-free pricing, because the parameter ρ tunes the properties of the utility function.

C. Key Modules

Now we start to discuss the design of three key modules in our framework depicted densely in Alg. 1. Because we should guarantee arbitrage-free pricing, Theorem 3.1 indeed guides our market design.

Algorithm 1 Trading with Controllable Privacy Loss

Input: privacy specification $\Phi = [\hat{\epsilon}_1, ..., \hat{\epsilon}_n]$, window specification $\mathcal{W} = [w_1, ..., w_n]$, compensation rates $[c_1, ..., c_n]$

$\tau \leftarrow 1$
while True **do**
  $[\hat{\epsilon}_1^\tau, ..., \hat{\epsilon}_n^\tau] \leftarrow$ **BudgetAlloc**()
  $[\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau], \rho \leftarrow$ **BudgetAdjusting**$([\hat{\epsilon}_1^\tau, ..., \hat{\epsilon}_n^\tau])$
  while not ($D^{\tau+1}$ is released) **do**
    Receive a query request Q^τ
    Compute the lower bound $\check{v}^\tau = U_{\mathcal{M}}([\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau])$
    Receive a histogram variance v^τ
    if $v^\tau < \check{v}^\tau$ **then**
      Reject Q^τ
    else
      Quote the buyer $\pi^\tau = \Pi_{\mathcal{M}}(v^\tau)$
      Collect the buyer's payment
      Calculate privacy losses: $[\epsilon_1^\tau, ..., \epsilon_n^\tau] \leftarrow {U_{\mathcal{M}}}^{-1}(v^\tau)$
      Compensate each data owner u_i according to $\mu_i(\epsilon_i^\tau)$
      Compute $\tilde{q}^\tau \leftarrow$ **PerturbMech**$(D^\tau, [\epsilon_1^\tau, ..., \epsilon_n^\tau])$
      Return back to the buyer the perturbed query answer $\tilde{q}^\tau$
      $[\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau] \leftarrow [\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau] - [\epsilon_1^\tau, ..., \epsilon_n^\tau]$
    end if
  end while
  $\tau \leftarrow \tau + 1$
end while

Before trading a perturbed query answer over D^1, the broker should make a contract with each data owner u_i to stipulate: the compensation rate c_i, the privacy loss bound $\hat{\epsilon}_i$, and the sliding-window size w_i. Then, two specifications which control the privacy protection level of personalized $\mathcal{W}$-event privacy (PWP) are derived: (1) the privacy specification $\Phi = [\hat{\epsilon}_1, ..., \hat{\epsilon}_n]$, and (2) the window specification $\mathcal{W} = [w_1, ..., w_n]$. Besides, the broker should decide which instances of the key modules to deploy.

PerturbMech: This module aims at perturbing query answers to achieve PWP. Although our framework should satisfy PWP, those perturbation mechanisms that satisfy personalized differential privacy (PDP) can be deployed as instances of *PerturbMech*. That is because, we propose Theorem 3.2 by which we can safely employ perturbation mechanisms satisfying Φ^τ-PDP at every time-point τ in order to achieve Φ-personalized $\mathcal{W}$-event privacy over the whole timeline. Hence, the task of achieving PWP on the timeline can be transformed into achieving PDP at every time-point.

Theorem 3.2 (Achieving PWP by PDP): Let $\mathcal{M}^1, ..., \mathcal{M}^\tau$ be a set of independent perturbation mechanisms where each $\mathcal{M}^t$ satisfies Φ^t-personalized differential privacy (where $\Phi^t = [\epsilon_1^t, ..., \epsilon_n^t]$). Let $\mathcal{M}$ be a mechanism which takes a stream prefix $S^\tau = (D^1, ..., D^\tau)$ as input and outputs $(\mathcal{M}^1(D^1, \Phi^1), ..., \mathcal{M}^\tau(D^\tau, \Phi^\tau))$. Then, $\mathcal{M}$ satisfies Φ-personalized $\mathcal{W}$-event privacy (where $\Phi = [\hat{\epsilon}_1, ..., \hat{\epsilon}_n]$) if for any $t \in [\tau]$ and any u_i, we have $\sum_{j=t-w_i+1}^{t} \epsilon_i^j \leq \hat{\epsilon}_i$.

Proof: Let $\mathcal{M}^t(D^t, \Phi^t) = o^t$ for all $t \in [\tau]$. Due to Definition 2.1, for any $t \in [\tau]$, for any $D^{t\prime}$ of D^t such that $D^t \overset{u_i}{\approx} D^{t\prime}$, we have:

$$Pr[\mathcal{M}_t(D^t) = o^t] \leq e^{\epsilon_i^t} Pr[\mathcal{M}_t(D^{t\prime}) = o^t].$$

Because each $\mathcal{M}^t$ computes o^t independently, we have:

$$Pr[\mathcal{M}(S^\tau) = o] = \prod_{t=1}^{\tau} Pr[\mathcal{M}^t(D^t) = o^t].$$

Then, for any w_i-neighboring stream prefix $S^{\tau\prime}$ of S^τ, there is $k \in [\tau]$ such that $D^t = D^{t\prime}$ for all $t \in [1, k - w_i] \cup [k+1, \tau]$. Therefore, we have:

$$\frac{Pr[\mathcal{M}(S^\tau) = o]}{Pr[\mathcal{M}(S^{\tau\prime}) = o]} = \frac{\prod_{t=k-w_i+1}^{k} Pr[\mathcal{M}^t(D^t) = o^t]}{\prod_{t=k-w_i+1}^{k} Pr[\mathcal{M}^t(D^{t\prime}) = o^t]} \leq \prod_{t=k-w_i+1}^{k} e^{\epsilon_i^t} = e^{\sum_{t=k-w_i+1}^{k} \epsilon_i^t} \leq e^{\hat{\epsilon}_i}.$$

Thus, $\mathcal{M}$ satisfies Φ-personalized $\mathcal{W}$-event privacy. ■

Besides, an instance of *PerturbMech* should guarantee the third property in Theorem 3.1. Intuitively, this property requires that, if data owners suffer no privacy loss, the utility should be extremely bad. Since whether this property can be satisfied naturally depends on the selection of perturbation mechanism $\mathcal{M}$, we should carefully design or select a perturbation mechanism $\mathcal{M}$ not only achieving PDP, but also guaranteeing this property.

BudgetAlloc: This module is designed for making full use of privacy loss bounds for data owners and making more queries answerable for data buyers. As the broker achieves PWP on the timeline by achieving PDP at every time-point, the task of budget allocation arises. As shown in Theorem 3.2, to satisfy Φ-personalized $\mathcal{W}$-event privacy, for each data owner u_i, the total privacy loss for every w_i successive time-points should be no more than her privacy loss bound, i.e., $\forall u_i, \forall t \in [\tau], \sum_{j=t-w_i+1}^{t} \epsilon_i^j \leq \hat{\epsilon}_i$. Thus, at every time-point τ, the broker should carefully allocate raw privacy budgets $[\hat{\epsilon}_1^\tau, ..., \hat{\epsilon}_n^\tau]$ such that for any u_i:

$$\sum_{j=\tau-w_i+1}^{\tau-1} \epsilon_i^j + \hat{\epsilon}_i^\tau \leq \hat{\epsilon}_i.$$

We can leverage those budget allocation techniques that support streaming scenarios, e.g., Budget Distribution and Budget Absorption proposed in [7], as instances of *BudgetAlloc*.

BudgetAdjusting: this module is a bond between *PerturbMech* and *BudgetAlloc*, designed for ensuring arbitrage-free pricing. Concretely, it takes over the role of making privacy budgets in ρ-pattern so as to satisfy the first and second properties in Theorem 3.1. Therefore, for every time-point τ, it always adjusts raw privacy budgets $[\hat{\epsilon}_i^\tau, ..., \hat{\epsilon}_i^\tau]$ and computes adjusted privacy budgets $[\bar{\epsilon}_i^\tau, ..., \bar{\epsilon}_i^\tau]$ in a fixed ρ-pattern, which results in a utility function satisfying the first two properties. That means the pattern ρ largely determines whether the

pricing function is arbitrage-free or not. Intuitively, if the price decreases rapidly as the histogram variance increases, it might be not worthwhile to purchase a query answer with a relatively low histogram variance v, which is not arbitrage-free. Hence, to make it worthwhile, an instance of *BudgetAdjusting* should provide a pattern ρ such that $\pi = \Pi_{\mathcal{M}}(v)$ decreases slowly enough as v increases, as implied mathematically by the first two properties in Theorem 3.1.

IV. Arbitrage-free Trading Protocols

In this section, we propose two arbitrage-free trading protocols by instantiating three key modules in Alg. 1: *BudgetAlloc*, *BudgetAdjusting*, and *PerturbMech*.

A. Baseline: UniformTrading

We propose the UniformTrading protocol as our baseline, which is a combination of the Minimum Laplace mechanism (as an instance of *PerturbMech*), Uniform (as an instance of *BudgetAdjusting*), and an arbitrary instance of *BudgetAlloc*.

PerturbMech: The Laplace mechanism is often used for implementing differential privacy and its variants. In the previous work [2], the broker trades query answers perturbed by the Laplace mechanism which causes uniform privacy losses for all data owners. In this paper, even though the upper bound of privacy loss is controlled by each data owner, which results in personalized privacy loss bounds, we can still deploy the Minimum Laplace mechanism (a variant of the Laplace mechanism) to perturb query answers.

BudgetAdjusting: However, according to Theorem 4.1, because the Minimum Laplace mechanism always controls the privacy protection level by the minimum element of $[\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau]$, it is waste to allocate adjusted privacy budgets $\bar{\epsilon}_i^\tau > \min_j \bar{\epsilon}_j^\tau$. Thus, we use Uniform to allocate uniform adjusted privacy budgets for the Minimum Laplace mechanism.

Theorem 4.1 (Minimum Laplace Mechanism [6]): Given a histogram query $f : \mathcal{D} \to \mathcal{N}^d$, a trajectory database D, and adjusted privacy budgets $[\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau]$, the Minimum Laplace mechanism which returns $f(D) + \mathcal{Z}^d$ satisfies Φ-PDP, where $\mathcal{Z}^d$ is a vector of random variables drawn from the Laplace distribution $Lap(\frac{\Delta_f}{\min_i \bar{\epsilon}_i^\tau})$ ($\Delta_f = \max_{D \sim D'} \|f(D) - f(D')\|_1$), and $\Phi = [\min_i \bar{\epsilon}_i^\tau, ..., \min_i \bar{\epsilon}_i^\tau]$.

Definition 4.1 (Uniform): Uniform allocates $[\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau] = [\min_i \hat{\epsilon}_i^\tau, ..., \min_i \hat{\epsilon}_i^\tau]$.

We can easily observe that $[\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau]$ allocated by Uniform is in ρ^{uni}-pattern where $\rho^{uni} = [1, ..., 1]$, and the utility function $U_{Lap}(\rho^{uni} \cdot \epsilon_{max}) = 2 \cdot (\frac{\Delta f}{\epsilon_{max}})^2$ satisfies the three properties in Theorem 3.1. Thus, $\pi = \Pi_{Lap}(v)$ is arbitrage-free. However, UniformTrading is not ideal for data owners because it cannot make full use of privacy loss bounds.

B. Advanced: PersonalizedTrading

To make full use of privacy loss bounds and personalize privacy losses for data owner, we propose the PersonalizedTrading protocol which is the combination of the Sample mechanism as an instance of *PerturbMech*, and Patterning (with a pre-processing algorithm PatternSearch) as an instance of *BudgetAdjusting*, and an arbitrary instance of *BudgetAlloc*.

PerturbMech: We leverage the Sample mechanism, a PDP technique, for module *PerturbMech*. According to Theorem 4.2, under the Sample mechanism, each data owner u_i's privacy loss ϵ_i^τ can be personalized.

Theorem 4.2 (Sample Mechanism [6]): Given a histogram query $f : \mathcal{D} \to \mathcal{N}^d$, a trajectory database D, and adjusted privacy budgets $[\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau]$, the Sample mechanism computes $\tilde{D}$ by sampling each row $D[i]$ with probability $Pr_i = (e^{\bar{\epsilon}_i^\tau} - 1)/(e^{\max_j \bar{\epsilon}_j^\tau} - 1)$, and returns $f(\tilde{D}) + \mathcal{Z}^d$, where $\mathcal{Z}^d$ is a vector of random variables drawn from the Laplace distribution $Lap(\frac{\Delta_f}{\max_j \bar{\epsilon}_j^\tau})$ ($\Delta_f = \max_{\tilde{D} \sim \tilde{D}'} \left\| f(\tilde{D}) - f\left(\tilde{D}'\right) \right\|_1$). The Sample mechanism satisfies Φ-PDP privacy, where $\Phi = [\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau]$.

BudgetAdjusting: We propose an algorithm named Patterning (Alg. 2) to adjust raw privacy budgets so that the adjusted privacy budgets are in a ρ-pattern, depicted as follows.

Algorithm 2 Patterning

Input: $[\hat{\epsilon}_1^\tau, ..., \hat{\epsilon}_n^\tau]$, ρ (pre-computed by Alg. 3)
Output: $[\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau]$

```
ε_max ← max_i ε̂_i^τ
for i = 1 to n do
    if ε̂_i^τ < ε_max * ρ_i then
        ε_max ← ε̂_i^τ / ρ_i
    end if
end for
return [ε̄_1^τ, ..., ε̄_n^τ] ← ε_max * ρ
```

Then, we should decide a ρ to adjust raw privacy budgets. An ideal solution is to make the pattern ρ of adjusted privacy budgets $[\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau]$ the same as the pattern ρ^{init} of the vector $[\frac{\hat{\epsilon}_1}{w_1}, ..., \frac{\hat{\epsilon}_n}{w_n}]$, i.e., $\rho = \rho^{init} = [\frac{\hat{\epsilon}_1}{w_1}, ..., \frac{\hat{\epsilon}_n}{w_n}] / \max_i \frac{\hat{\epsilon}_i}{w_i}$, by which the broker can make full use of all the data owners' privacy loss bounds $\hat{\epsilon}_i$. Unfortunately, for some ρ, the utility function $v = U_{Sam}(\rho \cdot \epsilon_{max})$ cannot satisfy the first two properties in Theorem 3.1, and therefore corresponds to a not arbitrage-free pricing function. For example, given a ρ, $v = U_{Sam}(\rho \cdot \epsilon_{max})$ might decrease too slowly as ϵ_{max} increases; in other words, ϵ_{max} and also the price $\pi = \Pi_{Sam}(v)$ might decrease too quickly as v increases, which allows arbitrage behaviors. Besides, because the utility function of the Sample mechanism is complex, i.e., $U_{Sam}(\rho \cdot \epsilon_{max}) = \sum_i Pr_i \cdot (1 - Pr_i) + 2 \cdot (\frac{\Delta f}{\max_j \epsilon_j})^2$ where $Pr_i = \frac{e^{\rho_i \cdot \epsilon_{max}} - 1}{e^{\epsilon_{max}} - 1}$, finding a pattern ρ which results in an arbitrage-free pricing function is extremely difficult. In fact, we even cannot express the inverse function ${U_{Sam}}^{-1}$ in analytic form.

However, given Theorem 3.1, we can transform the problem of finding a ρ resulting in an arbitrage-free pricing function under the Sample mechanism, into the following optimization problem:

$$\min_{\rho} \sum_i (\rho_i - \rho_i^{init})^2,$$

$$s.t.\ \forall \epsilon_{max} > 0,\ {U_{Sam}}'(\rho \cdot \epsilon_{max}) + \beta \leq 0,\ and$$

$$U_{Sam}(\rho \cdot \epsilon_{max}) \cdot {U_{Sam}}''(\rho \cdot \epsilon_{max}) - 2[{U_{Sam}}'(\rho \cdot \epsilon_{max})]^2 \leq 0.$$

The constraints are based on the first two properties in Theorem 3.1, where $\beta > 0$ is an extremely small positive

constant, and we skip the third property because it is always satisfied under the Sample mechanism. We also propose a heuristic named PatternSearch (Alg. 3) to solve this hard problem by binary search, based on the observation that for each element $\rho_i \neq 1$, the smaller the ρ_i is, the closer the utility function $v = U_{Sam}(\rho \cdot \epsilon_{max})$ is to satisfying the optimization constraints. We note that there must be such a ρ found by PatternSearch satisfying those constraints, because in the worst case where each element ρ_i is equal to 0 or 1, the utility function is $v = U_{Sam}(\rho \cdot \epsilon_{max}) = 2 \cdot (\frac{\Delta f}{\max_j \epsilon_j})^2$, which satisfies the three properties in Theorem 3.1.

Algorithm 3 PatternSearch

Input: ρ^{init}
Output: ρ
$\rho, \rho^{start}, \rho^{end} \leftarrow \rho^{init}$
For each $\rho_i^{end} \neq 1$, let $\rho_i^{end} = 0$
while True **do**
  $\rho^{pre} \leftarrow \rho$
  if optimization constraints satisfied **then**
    $\rho^{end} \leftarrow \rho$
    For each $\rho_i \neq 1$, let $\rho_i = (\rho_i + \rho_i^{start})/2$
  else
    $\rho^{start} \leftarrow \rho$
    For each $\rho_i \neq 1$, let $\rho_i = (\rho_i + \rho_i^{end})/2$
  end if
  if $\sum_i (\rho_i - \rho_i^{pre})^2 <$ a extremely small positive constant **then**
    return ρ^{end}
  end if
end while

C. Budget Allocation

We propose a budget allocation algorithm named Seize-the-Moment as an instance of module *BudgetAlloc*. Besides, the well-known budget allocation algorithm Budget Absorption (BA) is also leveraged but tailored to the needs of our settings.

Algorithm 4 Seize-the-Moment

Input: $\hat{\epsilon}_i$, $[\hat{\epsilon_i^1}, ..., \hat{\epsilon}_i^{\tau-1}]$, $[\epsilon_i^1, ..., \epsilon_i^{\tau-1}]$
Output: $\hat{\epsilon}_i^\tau$
$losses \leftarrow \sum_{t=\tau-w_i+1}^{\tau-1} \epsilon_i^t$ //w_i is a global constant.
$count \leftarrow$Calculate the times $\hat{\epsilon}_i^t = \epsilon_i^t$ for $t = 1$ to $\tau - 1$
$pro_S \leftarrow 1.0$ if $\tau = 1$, or $1.0 - 0.5 * (\frac{count}{\tau - 1})$ otherwise;
return $(\hat{\epsilon}_i - losses) * pro_S$

Seize-the-Moment (Alg. 4): The Seize-the-Moment algorithm always allocates each data owner's privacy loss as much as possible at every time-point. At time-point 1, it allocates all the privacy loss bounds as raw privacy budgets, i.e., $\hat{\epsilon}_i^1 = \hat{\epsilon}_i$. Then at every time point τ, for each data owner u_i, once a data buyer exhausts her privacy budget $\hat{\epsilon}_i^\tau$, Seize-the-Moment will decrease the proportion pro_S. If u_i's raw privacy budgets are used up all the time, the pro_S for u_i will be 0.5.

Budget Absorption (Alg. 5): Under BA, the privacy budgets remained at the previous time-point can be absorbed as a supplement to the current privacy budgets. Concretely, at every time-point τ, for each data owner u_i, at first BA allocates $\hat{\epsilon}_i / w_i$, and then additionally allocates the remained privacy budget $\hat{\epsilon}_i^{\tau-1} - \epsilon_i^{\tau-1}$. Thus, the raw privacy budget $\hat{\epsilon}_i^\tau$ can be equal to $\hat{\epsilon}_i / w_i + \hat{\epsilon}_i^{\tau-1} - \epsilon_i^{\tau-1}$. However, if the remaining privacy loss bound $\hat{\epsilon}_i - \sum_{t=\tau-w_i+1}^{\tau-1} \epsilon_i^t$ is less than $\hat{\epsilon}_i / w_i + \hat{\epsilon}_i^{\tau-1} - \epsilon_i^{\tau-1}$, it just allocates $\hat{\epsilon}_i^\tau = \hat{\epsilon}_i - \sum_{t=\tau-w_i+1}^{\tau-1} \epsilon_i^t$. The original

Algorithm 5 Budget Absorption

Input: $\hat{\epsilon}_i$, $[\hat{\epsilon}_i^1, ..., \hat{\epsilon}_i^{\tau-1}]$, $[\epsilon_i^1, ..., \epsilon_i^{\tau-1}]$
Output: $\hat{\epsilon}_i^\tau$
$\hat{\epsilon}_i^\tau \leftarrow \hat{\epsilon}_i / w_i$ //w_i is a global constant
if $\tau > 1$ **then**
  $absorption \leftarrow \hat{\epsilon}_i^\tau + \hat{\epsilon}_i^{\tau-1} - \epsilon_i^{\tau-1}$
  $remaining\_privacy\_loss\_bound \leftarrow \hat{\epsilon}_i - \sum_{t=\tau-w_i+1}^{\tau-1} \epsilon_i^t$
  $\hat{\epsilon}_i^\tau \leftarrow min(absorption, remaining\_privacy\_loss\_bound)$
end if
return $\hat{\epsilon}_i^\tau$

version of BA has a dissimilarity calculation sub-mechanism to compute the distance dis_1 between the raw query answer at time-point τ and a perturbed query answer published before τ, i.e., q^τ and $\tilde{q}^{pre}$. If dis_1 is lower than the expected distance dis_2 between the perturbed query answer and the raw query answer at time-point τ, i.e., $\tilde{q}^\tau$ and q^τ, then it will publish $\tilde{q}^{pre}$ instead of $\tilde{q}^\tau$. One limitation of the original BA is that it should deploy the Laplace mechanism as the perturbation mechanism. Besides, because it also perturbs the value of the distance dis_1, it might publish a $\tilde{q}^{pre}$ which is distant from q^τ. Thus, even though the dissimilarity calculation sub-mechanism provides better utility over the whole timeline, the utility might be worse for a single time-point. To provide a worst-case utility guarantee for data buyers, we drop this sub-mechanism.

V. Experiments

In this section, we simulate transactions in our data marketplace on synthetic data. We conduct two parts of experiments as follows: (1) from data owner' point of view, to verify the effectiveness of budget allocation algorithms from the aspect of making full use of data owners' privacy loss bounds, and (2) from data buyers' point of view, to verify the effectiveness of two arbitrage-free trading protocols in terms of making more queries answerable and improving the cost-performance.

Experiment setup: We generate n (the default n is 200) data owners' locations for 100 time-points. In this paper, since we do not consider any data correlation (which will be considered in our future work), those locations are randomly picked and represented by $d = 20$ regions for simplicity. Then, according to our previous user survey on privacy preference [5], we randomly divided those data owners into four groups: *conservative*, consisting of 16 percent of data owners with average privacy loss bounds $\hat{\epsilon}_i / w_i \in [0.01, 0.2]$; *hesitant*, consisting of 16 percent with $\hat{\epsilon}_i / w_i \in [0.2, 0.5]$; *ordinary*, consisting of 33 percent with $\hat{\epsilon}_i / w_i \in [0.5, 0.9]$; *liberal*, consisting of 34 percent with $\hat{\epsilon}_i / w_i = 1.0$. Besides, each data owner's preference on the sliding-window size w_i is uniformly at random picked from $[W - 1, W - 1]$ (the default W is 6).

A. Experiments for Data Owners

Now we start to verify the effectiveness of different budget allocation algorithms as instances of module *BudgetAlloc* in terms of making full use of privacy loss bounds for

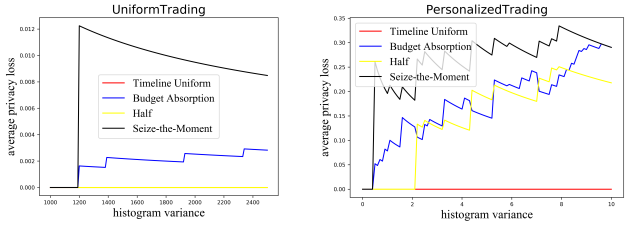

Fig. 4. Requesting the same histogram variance v

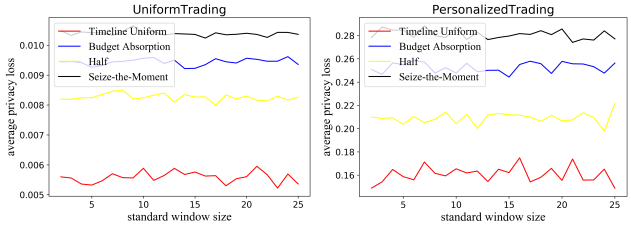

Fig. 5. Varying the standard sliding-window size

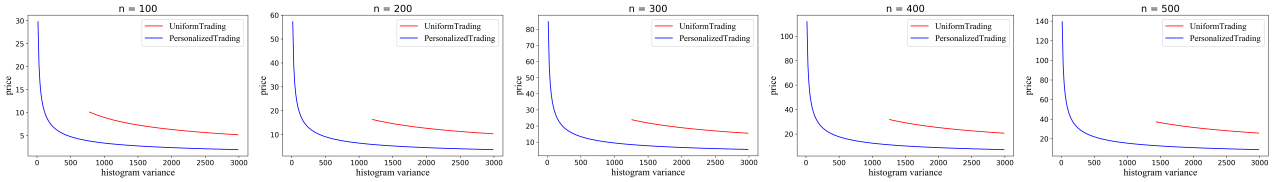

Fig. 6. Varying the number of data owners

data owners. We use two budget allocation algorithms as benchmarks: *Timeline Uniform* [7] which uniformly allocates privacy budgets $\hat{\epsilon}_i^\tau = \hat{\epsilon}_i / w_i$ for each time-point τ, and *Half*, which allocates $\hat{\epsilon}_i^\tau = (\hat{\epsilon}_i - \sum_{j=\tau-w_i+1}^{\tau-1} \epsilon_i^j)/2$ (i.e., half of the remaining privacy loss bounds).

Requesting the same histogram variance v: First, we simulate the cases where the data buyer requests the same histogram variance for all time-points. Those cases might be common because query answers of different time-points but with the same utility are useful for observing the dynamic changes of data owners' locational distribution. Figure 4 shows the average privacy loss consumed per time-point and per data owner under different instances. For both UniformTrading and PersonalizedTrading, if the data buyer requests a low histogram variance, Seize-the-Moment always outperforms the others because it can help sell higher privacy losses for data owners on average. However, with the histogram variance increasing, the gap of performance between those algorithms narrows. Hence, if data buyers always request relatively accurate queries, Seize-the-Moment will be superior. Timeline Uniform performs the worst where even no privacy loss is sold when the histogram variance is extremely low, because such queries are not answerable. Besides, higher privacy losses on average are sold under PersoanlizedTrading, which implies it performs better than UniformTrading in terms of making full use of privacy loss bounds.

Varying the standard sliding-window size W: Then we want to know the impact of the parameter W which controls the range of each data owner's preferred sliding-window size w_i. At each time-point τ, a data buyer randomly chooses a histogram variance v^τ higher than the lower bound $\check{v}^\tau$, which means that the query must be answerable. As shown in Figure 5, it seems that the sliding-window size has only a slight impact on the average privacy loss consumed. Besides, Seize-the-moment still performs the best for all the standard sliding-window sizes W for both two protocols.

B. Experiments for Data Buyers

Next, we evaluate PersonalizeTrading and UniformTrading on the cost-performance and the performance of making more queries answerable for data buyers, both employing Seize-the-Moment for *BudgetAlloc* due to its great performance.

Varying the number of data owners n: As depicted in Figure 6, PersonalizedTrading can always make more queries answerable because the lower bound of histogram variance under PersonalizedTrading is much lower than under UniformTrading; its cost-performance is also better because for the same histogram variance, the price is lower. As n increases, UniformTrading performs even worse, while the performance of PersonalizedTrading stays outstanding. That is because the Laplace mechanism (i.e., the perturbation mechanism for UniformTrading) only uses the minimum privacy loss to perturb a query answer, so that UniformTrading is extremely sensitive to the presence of those conservative data owners. In conclusion, UniformTrading performs poorly in terms of making more queries answerable and improving the cost-performance.

VI. Related Work

Personal Data Trading: Balazinska et al. [9] guide the tendency of the research on *Data Market* for the database community. Generally speaking, data privacy and data pricing are the key issues in the personal data trading research. To solve the data privacy issue, some works like [10], [11] employed cryptographic techniques to encrypt sensitive information when setting up a data marketplace; other works like [1]–[3] used perturbation techniques (e.g., differential privacy [8]) to add noise to the raw data. Since differential privacy naturally provides a convincing metric for quantifying privacy, i.e., privacy loss ϵ, in recent years, some efforts have been put into pricing data based on data owners' privacy loss. Specifically, Ghosh et al. [1] initiated the study of trading statistics over private data at auctions, perceiving the privacy loss as a commodity. Then, Koutris et al. [12] proposed the original model of query-based pricing where the query price is automatically computed by a pricing function. Li et al. [2] also adopted such a query-based pricing model with the constraints from arbitrage-free pricing and differential privacy. Then, Niu et al. [3] proposed a privacy-preserving framework for trading query answers over correlated data. However, the privacy loss can be infinite in their frameworks [2], [3]. Recently, Nget et

al. [5] analyzed data owners' privacy preferences and found that most of them value privacy more important than money. Hence in this paper, to achieve limited privacy losses, we allow each data owner to set an upper bound of her own privacy loss. Jung et al. [13] proposed a negotiation technique to let data owners and data buyers determine the value of privacy loss and its price through negotiations, without the guarantees of data utility and arbitrage-free pricing. As for mobile data, a decision-making framework was proposed by Aly et al. [14] to help data buyers decide a single location data point to purchase, while we consider trading query answers over infinite trajectory data streams rather than trading a raw trajectory data point.

Differentially Private Streaming Data Publication: The research on differentially private data publication in streaming settings was initiated by the works of Dwork et al. [15], [16]. They introduced two fundamental privacy notions for streaming scenarios: event-level privacy which hides a single event (e.g., a single trajectory data point in our settings), and user-level privacy which hides any user's data in the stream (e.g., the whole trajectory of a user). To protect a user's multiple events in a w-length sliding window, Kellaris et al. [7] introduced the notion of *w-event privacy* which bridges the gap between event-level privacy and user-level privacy. They also proposed two data publication mechanisms, named budget distribution (BD) and budget absorption (BA), which skip the current data publication if it can be accurately approximated by the last publication. The budget allocation strategies in both BD and BA can be deployed in our trading framework. Wang et al. [17] further proposed an adaptive budget allocation algorithm which dynamically allocates privacy budgets to increase the utility of the data publication. Then, in order to enable data owners to personalize the sliding-window size w in w-event privacy so that the privacy protection levels for them can be different, Cao et al. [18] extended w-event privacy to l-trajectory privacy. Also, they investigated data correlations in streaming data which cause temporal privacy losses, and proposed mechanisms to bound such privacy losses [19] [20]. We combine w-event privacy and personalized differential privacy together to publish query answers over infinite streaming data, so that both the sliding-window size and privacy losses can be personalized for data owners.

VII. Conclusion

We proposed a framework for trading infinite streaming mobile data where each data owner's privacy loss under personalized $\mathcal{W}$-event privacy is bounded. To avoid arbitrage behaviors, we proposed Theorem 3.1 presenting the sufficient condition for arbitrage-free pricing, which guided the design of our trading framework. Then, based on this theorem, we further designed two arbitrage-free trading protocols by instantiating the three key modules in our framework. The PersonalizedTrading protocol outperformed the UniformTrading protocol in terms of making full use of privacy loss bounds for data owners, making more queries answerable and improving the cost-performance for data buyers.

VIII. Acknowledgement

This work is partially supported by JSPS KAKENHI Grant No. 17H06099, 18H04093, 19K20269, and Microsoft Research Asia (CORE16).

References

[1] A. Ghosh and A. Roth, "Selling privacy at auction," in *Proceedings of the 12th ACM conference on Electronic commerce*, 2011, pp. 199–208.

[2] C. Li, D. Y. Li, G. Miklau, and D. Suciu, "A theory of pricing private data," *ACM Transactions on Database Systems (TODS)*, vol. 39, no. 4, pp. 1–28, 2014.

[3] C. Niu, Z. Zheng, F. Wu, S. Tang, X. Gao, and G. Chen, "Unlocking the value of privacy: Trading aggregate statistics over private correlated data," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 2031–2040.

[4] L. Chen, P. Koutris, and A. Kumar, "Towards model-based pricing for machine learning in a data marketplace," in *Proceedings of the 2019 International Conference on Management of Data*, 2019, pp. 1535–1552.

[5] R. Nget, Y. Cao, and M. Yoshikawa, "How to balance privacy and money through pricing mechanism in personal data market," *arXiv preprint arXiv:1705.02982*, 2017.

[6] Z. Jorgensen, T. Yu, and G. Cormode, "Conservative or liberal? personalized differential privacy," in *2015 IEEE 31St international conference on data engineering*. IEEE, 2015, pp. 1023–1034.

[7] G. Kellaris, S. Papadopoulos, X. Xiao, and D. Papadias, "Differentially private event sequences over infinite streams," *Proceedings of the VLDB Endowment*, vol. 7, no. 12, pp. 1155–1166, 2014.

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

[9] M. Balazinska, B. Howe, and D. Suciu, "Data markets in the cloud: An opportunity for the database community," *Proc. of the VLDB Endowment*, vol. 4, no. 12, pp. 1482–1485, 2011.

[10] L. Zhang, J. Ren, Y. Mu, and B. Wang, "Privacy-preserving multi-authority attribute-based data sharing framework for smart grid," *IEEE Access*, vol. 8, pp. 23 294–23 307, 2020.

[11] C. Niu, Z. Zheng, F. Wu, X. Gao, and G. Chen, "Trading data in good faith: Integrating truthfulness and privacy preservation in data markets," in *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*. IEEE, 2017, pp. 223–226.

[12] P. Koutris, P. Upadhyaya, M. Balazinska, B. Howe, and D. Suciu, "Query-based data pricing," in *Proceedings of the 31st ACM SIGMOD-SIGACT-SIGAI symposium on Principles of Database Systems*, 2012, pp. 167–178.

[13] K. Jung and S. Park, "Privacy bargaining with fairness: Privacy-price negotiation system for applying differential privacy in data market environments," in *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, 2019, pp. 1389–1394.

[14] H. Aly, J. Krumm, G. Ranade, and E. Horvitz, "On the value of spatiotemporal information: Principles and scenarios," in *Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 2018, pp. 179–188.

[15] C. Dwork, M. Naor, T. Pitassi, and G. N. Rothblum, "Differential privacy under continual observation," in *Proceedings of the forty-second ACM symposium on Theory of computing*, 2010, pp. 715–724.

[16] C. Dwork, "Differential privacy in new settings," in *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*. SIAM, 2010, pp. 174–183.

[17] Q. Wang, Y. Zhang, X. Lu, Z. Wang, Z. Qin, and K. Ren, "Real-time and spatio-temporal crowd-sourced social network data publishing with differential privacy," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, pp. 591–606, 2016.

[18] Y. Cao and M. Yoshikawa, "Differentially private real-time data release over infinite trajectory streams," in *2015 16th IEEE International Conference on Mobile Data Management*, vol. 2. IEEE, 2015, pp. 68–73.

[19] Y. Cao, M. Yoshikawa, Y. Xiao, and L. Xiong, "Quantifying differential privacy under temporal correlations," in *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*. IEEE, 2017, pp. 821–832.

[20] ——, "Quantifying differential privacy in continuous data release under temporal correlations," *IEEE transactions on knowledge and data engineering*, vol. 31, no. 7, pp. 1281–1295, 2018.