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

Shuyuan Zheng
Graduate School of Informatics
Kyoto University
Kyoto, Japan
caryzheng@db.soc.i.kyoto-u.ac.jp

Yang Cao
Graduate School of Informatics
Kyoto University
Kyoto, Japan
yang@i.kyoto-u.ac.jp

Masatoshi Yoshikawa
Graduate School of Informatics
Kyoto University
Kyoto, Japan
yoshikawa@i.kyoto-u.ac.jp

***Abstract*—As personal data have 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. It is assumed that a data buyer could purchase arbitrarily accurate results as long as she could compensate data owners' privacy loss. However, it may not be true in reality since our recent survey reveals that most of the data owners valued privacy more than money. In this paper, we study how to empower data owners with the control of privacy loss when continuously trading their personal mobile data. Specifically, we propose a framework for trading infinite streaming mobile data that enables each data owner to bound her privacy loss in any w sliding window. Introducing such upper bounds of privacy loss makes the existing trading frameworks invalid and raises new technical challenges in the aspects of budget allocation and arbitrage-free pricing. To address these problems, we propose a modularized trading framework with instances that allow 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 protocols.**

***Index Terms*—personal data trading, controllable privacy loss, personalized differential privacy, budget allocation, arbitrage-free pricing**

I. Introduction

Personal data, the new oil of the digital era, are extraordinarily valuable for individuals and organizations to discover knowledge and improve products or services. While a single individual's personal information is worth nothing in practice, its aggregate can be worth billions [1]. However, since personal data may also release sensitive information which can be used to identify individuals for crimes, individuals also deserve appropriate compensations due to their potential *privacy loss*. In particular, a study found that a compensation, especially monetary one, reduces people' expectations for privacy protection [2], which implies that some individuals would like to provide personal data in exchange of money. In fact, there is a growing trend towards *personal data trading* perceiving personal data as a commodity, which meets the demand of both data buyers (i.e., who want to utilize personal data) and data owners (i.e., who generate the data). Some startup companies, such as Datacoup[1] and CitizenMe[2], consider personal data trading platform that connects data owners and data buyers directly as a new business model. This model allows data owners to trade off privacy loss for money, which benefits personal data sharing and utilization.

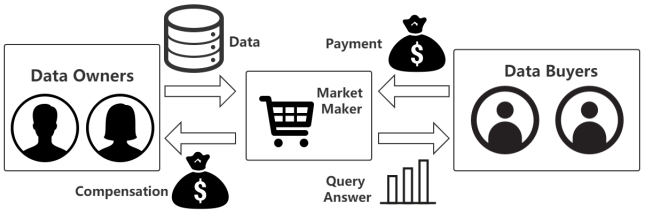

Fig. 1. Query-based Data Trading

Several studies [3]–[8] 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 *market maker*. Data owners contribute their personal data and get monetary compensations from the market maker in return. Data buyers request queries over the data and pay for perturbed query answers, i.e., noisy versions of aggregate statistical results where some random noises are injected. The market maker acts as a trustworthy intermediary between data owners and data buyers, in charge of computing perturbed query answers, setting *query price* for data buyers and compensating data owners. A major challenge in the line of works is how to price a query answer. A seminal work of Li et al. [6] assigned query prices to queries according to their utility. They designed pricing functions by making the connection between privacy loss and the utility, and formulated an important property of pricing functions: *arbitrage-freeness*, which means the consistency of a set of priced queries. Intuitively, a buyer should not obtain the answer to a query by deriving this answer from a less expensive set of queries.

Problems of Previous Works

However, the previously proposed data markets did not give the power to data owners to control their privacy loss in continuous data trading. Concretely, there are several desiderata in a personal data market that have not been well studied. First, data owners should be able to specify the *upper bound* of the

[1]http://datacoup.com/
[2]https://citizenme.com/

privacy loss given the high sensitivity of mobile data such as location trajectories. In the traditional data marketplace shown in Figure 1, a data buyer even can purchase query answers with arbitrary accuracy (theoretically, it is equivalent to purchase the raw data) as long as she could appropriately pay for it. However, our recent survey [9] reveals that most of the data owners valued 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 *personalized* privacy loss bound she like; however, the existing studies [3] [6] [7] enforce a uniform privacy loss for all data owners. Third, existing studies mainly focus on trading static personal data; however, a large amount of personal data, such as streaming mobile data, are generated *continuously*.

Challenges

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 utility in a data marketplace. There is no framework in the existing studies for arbitrage-free pricing under utility constraints. This lead to a new and challenging problem: how to price query answers based on their utility in an arbitrage-free manner.

Second, personalized privacy losses make it more difficult to design an arbitrage-free pricing function. As query price depends on the utility of query answer 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 means 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; otherwise, it might be too complex to express it in analytic form, study the mathematical properties of both itself and its corresponding pricing function, and improve it to guarantee arbitrage-free pricing. Although we can leverage *personalized differential privacy* (PDP) [10] 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 market maker should protect every trajectory, the task of privacy budget allocation raises, concerning not only the privacy losses at a single time-point, but also on the timeline. Although w-event privacy [11] 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 are extremely complicated.

Contributions

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 the supply (i.e., bounded privacy loss from data owners) and demand (i.e., utility requirement from data buyers) in a data market. The major challenge is how to ensure arbitrage-free under bounded utility. Essentially, it requires the market maker to derive the privacy loss given a required utility (i.e., variance), which is non-trivial if the differentially private mechanism is complicated or the privacy loss is personalized. We present a general method to design an arbitrage-free pricing function based on the inverse function of the utility function of query answers, and propose a method to adjust the given privacy budgets from data owners so that 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 the protocol 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 function. 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 whole view of our trading framework. Section 4 discusses the technical details in the modules of our framework. Section 5 presents the experimental results. Section 6 introduces the related work 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 and $D^\tau[i] \in \{1, ..., d\}$ represents data owner u_i's location ID number at the 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 an histogram query Q^τ over the trajectory database D^τ. A trajectory database D^τ can be transformed into the raw query answer $q^\tau = (q_1^\tau, ..., q_d^\tau)^T$ where the element q_l^τ is the total number of data owners who was in the location of ID l at the time-point τ.

Personalized differential privacy (PDP): To preserve data owners' privacy, the raw query answer cannot be returned to the buyer and the market maker should perturb it by some perturbation mechanism $\mathcal{M}$ achieving a formal privacy standard. Since data owners are allowed to set upper bound of individual privacy loss in our settings, we deploy personalized differential privacy (a variation of differential privacy [12]), where each data owner u_i corresponds to a personalized privacy protection level, i.e., a privacy loss ϵ_i. [10] proposed Sample Mechanism to achieve PDP, and showed that Laplace Mechanism [12] can also be used to achieve it but result in uniform privacy losses.

Definition 2.1 (Personalized Differential Privacy [10]): Given a privacy specification $\Phi = [\epsilon_1, ..., \epsilon_n]$, a perturbation mechanism $\mathcal{M} : \mathcal{D} \to \mathcal{R}^d$ ($d \in Z^+$) satisfies Φ-personalized differential privacy, if for any pair of neighboring databases $D, D' \subset D$, with $D \overset{u_i}{\sim} D'$, and for any possible output $o \in \mathcal{R}^d$, we have: $Pr[\mathcal{M}(D) = o] \leq e^{\epsilon_i} Pr[\mathcal{M}(D') = o]$.

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

Utility metric: We need a utility metric to evaluate perturbed query answers, i.e., histogram variance v. A perturbed query answer $\tilde{q^\tau}$ is a vector where each element $\tilde{q}_l^\tau$ represents the sum of data owners in the location of ID l at time-point τ. The histogram variance of $\tilde{q}^\tau$ should be always no less than the maximum variance among all the element in $\tilde{q}^\tau$, i.e., $v \geq \max_l Var(\tilde{q}_l^\tau)$. Variance has been used as utility metric in several privacy-preserving data marketplaces [6]–[8] since it reflects the expected dispersion of a random variable, i.e., 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 in a vector.

B. Market Setup

There are three parties of participants in our marketplace: data owners, data buyers, and a market maker (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 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 $\hat{\epsilon}_i$ she can tolerate at 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., the histogram variance v) until $D^{\tau+1}$ is released, and get a perturbed query answer $\tilde{q}^\tau = Q^\tau(D^\tau, v)$ such that $v = \max_l Var(\tilde{q}_l^\tau)$. 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 [10].

The market maker and data owners. For preserving data owners' privacy, the market maker is in charge of perturbing the raw query answer by some perturbation mechanism $\mathcal{M}$ which satisfies personalized differential privacy (PDP, see Definition 2.1). Since PDP only protects a single trajectory database over the entire stream, we present personalized $\mathcal{W}$-event privacy (see Definition 2.2) based on PDP [10] and w-event privacy [11] to guarantee that the privacy loss of each data owner's w_i successive data points is not exceeding $\hat{\epsilon}_i$.

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 pair of w_i-neighborbing stream prefixes S^τ, $S^{\tau\prime}$, with any pair of their neighboring elements $S^\tau[t] \overset{u_i}{\sim} S^{\tau\prime}[t]$, $t \in [\tau]$, any τ, and for possible output o, we have:

$$Pr[\mathcal{M}(S^\tau) = o] \leq e^{\epsilon_i} Pr[\mathcal{M}(S^{\tau\prime}) = o]$$

Two stream prefixes S^τ, $S^{\tau\prime}$ are w_i-neighboring if:

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

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

The market maker and the buyer. As a profit-making intermediary, the market maker sells perturbed histogram queries to make profit. In our settings, the price of a query answer depends directly on its histogram variance v, rather than on any individual privacy loss it causes, because data buyers care the utility of query answers more. The market maker should design a pricing function $\pi = \Pi(v)$ which is: (1) *cost-covering*, which means query price should cover the total compensations to data owners, and (2) *arbitrage-free*, which means a buyer cannot obtain a perturbed query answer with a specific histogram variance v more cheaply by deriving a new answer 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 property: for any perturbed query answer q with a histogram variance v which causes privacy loss ϵ_i for each data owner u_i, we always 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 the pricing function which covers a profit.

Definition 2.4 (Arbitrage-freeness): A pricing function $\pi = \Pi(v)$ is arbitrage-free, if it satisfies the following property: 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 always have $\Pi(v) \leq \sum_{j=1}^{m} \Pi(v_j)$.

Goals. We summarize 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 more compensations.
- *Data buyers.* Because individual privacy loss ϵ_i has a upper bound $\hat{\epsilon}_i$ for each data owner u_i, the histogram variance v also has a lower bound. Thus, a data buyer may fail to purchase a query answer with a very low histogram; one of our design goals is to lower such a lower bound for data buyers to make more queries answerable. Besides, we also try to raise the cost-performance, i.e., to lower query price for the same histogram variance.
- *The market maker.* The pricing function should be arbitrage-free and cost-covering.

III. Mobile Data Trading Framework

In this section, first we give a whole 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 provides a framework to compute arbitrage-free trading protocols. Some important notations are summarized in Table I.

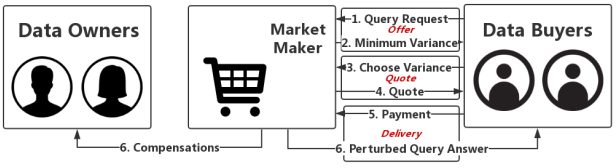

Fig. 2. Transaction Flow in our Trading Framework

A. Framework Overview

After the release of D^τ, a transaction of query answer in our data marketplace works as depicted in Figure 2. Before every transaction, the market marker should run two budget allocation modules briefly introduced as follows:

- BudgetAlloc (budget allocation): it allocates privacy budgets $[\hat{\epsilon}_1^\tau, ..., \hat{\epsilon}_n^\tau]$, with the goals of not only making full use of data owners' privacy losses within their privacy loss bounds but also making more queries answerable.

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 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
$\pi = \Pi_\mathcal{M}(v)$	pricing function

  We note that a privacy budget is the budget of privacy loss for a single time-point while a privacy loss bound is for multiple time-points.
- BudgetAdjusting (budget adjusting): it adjusts privacy budgets to adjusted privacy budgets $[\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau]$ in order to guarantee arbitrage-free pricing. We will discuss why we need to adjust privacy budgets in Section III-B. We note that each data owner's adjusted privacy budget $\bar{\epsilon}_i^\tau$ should be no more than privacy budget $\hat{\epsilon}_i^\tau$.

Once the adjusted privacy budgets are computed, the market maker starts to monitor query requests and trade perturbed query answers in the following steps.

Offer: In Step 1, a data buyer requests a query Q^τ over the database D^τ. Then in Step 2, the market maker will the maximum utility to the buyer, i.e., the lower bound $\check{v}^\tau$ of histogram variance. 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 $\check{v}^\tau$ depends on the adjusted privacy budgets $\bar{\epsilon}_i^\tau$, 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 buyer should choose a histogram variance $v^\tau \geq \check{v}^\tau$. Then in Step 4, the market maker quotes a price π^τ for a perturbed query answer $\tilde{q}^\tau = Q^\tau(D^\tau, v^\tau)$ by an arbitrage-free and cost-covering pricing function $\pi = \Pi(v)$.

Delivery: In Step 5, the buyer pays the price π^τ to the market maker. Then in Step 6, the market maker calculates each owner's privacy loss ϵ_i^τ for v^τ, computes a perturbed query answer $\tilde{q}^\tau = \mathcal{M}(D^\tau, [\epsilon_1^\tau, ..., \epsilon_n^\tau])$, returns $\tilde{q}^\tau$ to the buyer, and compensates each data owner according to her privacy loss. Each data onwer's privacy loss ϵ_i^τ really consumed should be no more than the 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 achieves 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 in our framework, we introduce how we design a pricing function.

Inverse function: Remind that in Step 6, given a histogram variance v^τ, the market maker 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 raises how to calculate the corresponding privacy losses given a histogram variance, which can be transformed into the problem of finding the mapping from the set of histogram variance to the set of privacy losses. In fact, because the utility function $v = U_\mathcal{M}([\epsilon_1, ..., \epsilon_n])$ actually 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 that ${U_\mathcal{M}}^{-1}$ 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., compensations to data owners. Remind that the pricing function should be cost-covering, i.e., $\pi = \Pi_\mathcal{M}(v) \geq \sum_i c_i \cdot \epsilon_i$. The direct way of designing the pricing function is to make 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 if the market maker is profit-making, she can modify the pricing function by adding 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 cases.

To make the inverse function existent: The inverse function under a simple perturbation mechanism such as Laplace Mechanism [12] could be easy to derive. However, it may not exist for a sophisticated mechanism such as Sample Mechanism [10]. 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 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 $\exists j$ such that $\rho_j = 1$. If a vector $[\epsilon_1, ..., \epsilon_n] = \rho \cdot \max_i \epsilon_i$, then we say 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 market maker allocates adjusted privacy budgets $[\bar{\epsilon}_1^1, \bar{\epsilon}_2^1, \bar{\epsilon}_3^1] = [4, 2, 2]$ and the buyer consumes privacy losses $[\epsilon_1^1, \epsilon_2^1, \epsilon_3^1] = [2, 1, 1]$; at time-point 2, $[\bar{\epsilon}_1^1, \bar{\epsilon}_2^2, \bar{\epsilon}_3^2] = [6, 3, 3]$ 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 ${U_\mathcal{M}}^{-1}(v)$ 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 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 functions are so complex that the inverse functions even cannot be expressed in analytic form, which makes it difficult to analyze the mathematical properties of the pricing functions and then guarantee arbitrage-free pricing. To solve this problem, considering whether we can study the properties of the utility function $U_\mathcal{M}(\rho \cdot \epsilon_{max})$ instead of $\Pi_\mathcal{M}$ to guarantee arbitrage-free pricing, we propose Theorem 3.1 illustrating that if $U_\mathcal{M}(\rho \cdot \epsilon_{max})$ satisfies some properties, then $\pi = \Pi_\mathcal{M}(v)$ 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) decreasing, which means $\forall \epsilon_{max} > 0$, ${U_\mathcal{M}}'(\rho \cdot \epsilon_{max}) < 0$;
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) $\lim_{\epsilon_{max} \to 0^+} U_\mathcal{M}(\rho \cdot \epsilon_{max}) = +\infty$.

where ${U_\mathcal{M}}'(\rho \cdot \epsilon_{max})$ and ${U_\mathcal{M}}''(\rho \cdot \epsilon_{max})$ are the first and second order derivatives of ϵ_{max}, correspondingly.

Thus, we can transform the problem of designing an arbitrage-free pricing function into find a utility function satisfying such properties. Intuitively, the first property makes the pricing function decreasing; the second guarantees a smooth decreasing speed; the third means a zero price corresponds to infinitely high variance. We note that, the choice of ρ is essentially the key to arbitrage-free pricing because the pattern ρ tunes the properties of the utility function $v = U_\mathcal{M}(\rho \cdot \epsilon_{max})$. The proofs of our theorems can be checked in [13].

C. Key Modules

Now we start to discuss the design of three key modules in our framework depicted densely in Alg. 1, and why we need them. Because we should guarantee arbitrage-free pricing by the sufficient condition proposed in Theorem 3.1, this theorem certainly guides our 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 $\pi = \Pi_{\mathcal{M}}(v^\tau)$;
      Collect the buyer's payment.
      Calculate privacy losses: $[\epsilon_1^\tau, ..., \epsilon_n^\tau] \leftarrow {U_{\mathcal{M}}}^{-1}(v^\tau)$;
      Compensate $\mu_i(\epsilon_i^\tau)$ to each data owner u_i;
      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 perturbed query answers over D^1, the market maker 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 $\Phi = [\hat{\epsilon}_1, ..., \hat{\epsilon}_1]$, (2) the window specification $\mathcal{W} = [w_1, ..., w_n]$. Besides, the market maker should decide which instances of three key modules to deploy.

PerturbMech: It is the module aiming at perturbing query answers to achieve PWP. Although our framework should satisfy PWP, perturbation mechanisms satisfying personalized differential privacy (PDP) can be used as instances of *PerturbMech*. That is, we propose Theorem 3.2 by which we can safely deploy perturbation mechanisms $\mathcal{M}_{\Phi_\tau}$ satisfying Φ_τ-PDP at every time-point τ to achieve Φ-personalized $\mathcal{W}$-event privacy on 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 $\mathcal{M}_{\Phi_1}, ..., \mathcal{M}_{\Phi_\tau}$ be a set of perturbation mechanisms where each independent sub-mechanism $\mathcal{M}_{\Phi_t}$ satisfies Φ_t-personalize differential privacy (where $\Phi_t = [\epsilon_1^t, ..., \epsilon_n^t]$), and $\mathcal{M}_{\Phi_t}(D^t) = o_t$. Let $\mathcal{M}$ be a mechanism which takes a stream prefix $S^\tau = (D^1, ..., D^\tau)$ as input and outputs $(\mathcal{M}_1(D^1), ..., \mathcal{M}_\tau(D^\tau))$. 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 $\sum_{j=t-w_i+1}^{t} \epsilon_i^j \leq \hat{\epsilon}_i$

Besides, an instance of *PerturbMech* should guarantee the third property in Theorem 3.1. Intuitively, this property requires that, intuitively, if data owners suffer no privacy loss, the utility should be infinitely bad. Since whether this property can be satisfied naturally depends on the choice of perturbation mechanism $\mathcal{M}$ and the choice of utility metric, 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 losses for data owners while making more queries answerable for data buyers. As the market maker achieves PWP on the timeline by achieving PDP at every time-point, the task of budget allocation raises. As shown in Theorem 3.2, to satisfy Φ-personalized $\mathcal{W}$-event privacy, for each data owner u_i, the total privacy losses $\sum_{j=t-w_i+1}^{t} \epsilon_i^j$ in the sliding window of every w_i successive time-points should be no more than her privacy loss bound $\hat{\epsilon}_i$. Thus, at every time-point τ, the market maker should carefully allocate privacy budgets $[\hat{\epsilon}_1^\tau, ..., \hat{\epsilon}_n^\tau]$ online such that for any u_i:

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

We can leverage those budget allocation techniques which support streaming data, e.g., Budget Distribution and Budget Absorption proposed in [11], as instances of *BudgetAlloc*.

BudgetAdjusting: BudgetAdjusting is a bond between *PerturbMech* and *BudgetAlloc*, designed for the purpose of deriving an arbitrage-free pricing function. In particular, it takes over the role of (1) making privacy budgets in ρ-pattern and then (2) guaranteeing the first and second properties in Theorem 3.1. Therefore, for every time-point τ, it always adjusts the 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 variance increases, it is more worthwhile to buy a query answer with high variance. Thus, to avoid arbitrage behaviors, an instance of *BudgetAdjusting* should find a pattern ρ such that ϵ_{max} always decreases smoothly to some extent as v increases and $\pi = \Pi_{\mathcal{M}}(v)$ decreases slow enough, as implied mathematically by the first two properties.

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 Minimum Laplace Mechanism (as an instance of *PerturbMech*), Uniform (as an instance of *BudgetAdjusting*), and an arbitrary instance of *BudgetAlloc*.

PerturbMech: Laplace Mechanism is a very basic and common mechanism for implementing differential privacy and its variations. In previous work [6], the market maker trades query answers perturbed by Laplace Mechanism, which results in unbounded uniform privacy losses for data owners. In this paper, though the upper bound of individual privacy loss is controlled by each data owner, which causes personalized upper bounds, we can still deploy Minimum Laplace Mechanism (a variation of Laplace Mechanism) to perturb queries.

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

Theorem 4.1 (Minimum Laplace Mechanism [10]): Given a histogram query $f : \mathcal{D} \to \mathcal{N}^d$, a trajectory database D, and adjusted privacy budgets $[\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau]$, Minimum Laplace Mechanism which returns $f(D) + \mathcal{Z}^d$ satisfies Φ-PDP, where $\mathcal{Z}^d$ are 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): Given privacy budgets $[\hat{\epsilon}_1^\tau, ..., \hat{\epsilon}_n^\tau]$, Uniform allocates $[\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau] = [\min_i \hat{\epsilon}_1^\tau, ..., \min_i \hat{\epsilon}_i^\tau]$.

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

In order to make full use of privacy loss bounds and personalize privacy loss for each data owner, we propose the PersonalizedTrading protocol which is the combination of 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 Sample Mechanism, one of PDP techniques, for module *PerturbMech*. According to Theorem 4.2, under Sample Mechanism, each u_i's privacy loss ϵ_i^τ can be personalized.

Theorem 4.2 (Sample Mechanism [10]): Given a histogram query $f : \mathcal{D} \to \mathcal{N}^d$, a trajectory database D, and adjusted privacy budgets $[\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau]$, 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 then returns $f(\tilde{D}) + \mathcal{Z}^d$, where $\mathcal{Z}^d$ are random variables drawn from the Laplace distribution $Lap(\frac{\Delta_f}{\max_j \bar{\epsilon}_j^\tau})$, where $\Delta_f = \max_{\tilde{D} \sim \tilde{D}'} \left\| f(\tilde{D}) - f\left(\tilde{D}'\right) \right\|_1$. Sample mechanism satisfies Φ-PDP privacy, where $\Phi = [\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau]$.

BudgetAdjusting: As for the choice of *BudgetAdjusting*, we propose a algorithm named Patterning (Alg. 2) to adjust privacy budgets so that the adjusted privacy budgets are in a ρ-pattern, depicted as follows.

Then, the question remains which ρ to adjust the 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 $[\frac{\hat{\epsilon}_1}{w_1}, ..., \frac{\hat{\epsilon}_n}{w_n}]$, i.e., $\rho^{init} = [\frac{\hat{\epsilon}_1}{w_1}, ..., \frac{\hat{\epsilon}_n}{w_n}] / \max_i \frac{\hat{\epsilon}_i}{w_i}$, by which the market maker can make full use of all the data owners' privacy loss bounds $\hat{\epsilon}_i$. Unfortunately, for some ρ, the utility functions $v = U_{Sam}(\rho \cdot \epsilon_{max})$ might violate the first two properties in Theorem 3.1, and result in not arbitrage-free pricing functions.

Algorithm 2 Patterning

Input: $[\hat{\epsilon}_1^\tau, ..., \hat{\epsilon}_n^\tau]$, ρ (pre-computed by Alg. 3)
Output: $[\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau]$
1: $\epsilon_{max} \leftarrow \max_i \hat{\epsilon}_i^\tau$;
2: **for** $i = 1$ **to** n **do**
3: **if** $\hat{\epsilon}_i^\tau < \epsilon_{max} * \rho_i$ **then**
4: $\epsilon_{max} \leftarrow \hat{\epsilon}_i^\tau / \rho_i$;
5: **end if**
6: **end for**
7: **return** $[\bar{\epsilon}_1^\tau, ..., \bar{\epsilon}_n^\tau] \leftarrow \epsilon_{max} * \rho$;

For example, in some ρ-patterns, $v = U_{Sam}(\rho \cdot \epsilon_{max})$ might decrease too slow as ϵ_{max} increases; in other words, ϵ_{max} and also the price $\pi = \Pi_{Sam}(v)$ might decrease too quick as v increases, which allows arbitrage behaviors. Worse yet, because the utility function of Sample Mechanism is sophisticated, 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 $Pr_i = \frac{e^{\rho_i \cdot \epsilon_{max}} - 1}{e^{\epsilon_{max}} - 1}$, it is also extremely difficult to find a pattern ρ resulting in an arbitrage-free pricing function. In fact, we even cannot express the inverse function ${U_{Sam}}^{-1}$ in analytic form.

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

$$\min_{\rho} \sum_i (\rho_i - \rho_i^{init})^2,$$
$$s.t.\ \forall \epsilon_{max} > 0,\ {U_{Sam}}'(\rho \cdot \epsilon_{max}) + \beta \leq 0,\ and$$
$$U_{Sam}(\rho \cdot \epsilon_{max}) \cdot {U_{Sam}}''(\rho \cdot \epsilon_{max}) - 2[{U_{Sam}}'(\epsilon_{max})]^2 \leq 0$$

The constraints are based on the first and second properties in Theorem 3.1, where $\beta > 0$ is an extremely small positive constant, and we drop the third property because it is always satisfied under Sample Mechanism. We also propose a algorithm named PatternSearch (Alg. 3) to solve this 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 $\rho_i \neq 1$ is equal to 0, $v = U_{Sam}(\rho \cdot \epsilon_{max}) = 2 \cdot (\frac{\Delta f}{\max_j \epsilon_j})^2$, which satisfies the three properties in Theorem 3.1.

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 fit our settings.

Seize-the-moment (Alg. 4): Seize-the-moment means it always allocates each data owner's privacy loss as much as possible at the moment. At time-point 1, it allocates privacy budgets with the proportion $pro_S = 1$. At the time point τ, for the owner u_i, once the buyer exhausts her privacy budget $\hat{\epsilon}_i^\tau$, Seize-the-moment decreases the proportion pro_S for her. If buyers use up u_i's privacy budgets all the time, the pro_S for u_i will be 0.5.

Algorithm 3 PatternSearch

Input: ρ^{init};
Output: ρ;
$\rho, \rho^{start}, \rho^{end} \leftarrow \rho^{init}$;
$\rho^{end} \leftarrow$ for each $\rho_i^{end} \neq 1$, let $\rho_i^{end} = 0$;
while True **do**
  $\rho^{pre} \leftarrow \rho$;
  if optimization constraints satisfied **then**
    $\rho^{end} \leftarrow \rho$;
    $\rho \leftarrow$ for each $\rho_i \neq 1$, let $\rho_i = (\rho_i + \rho_i^{start})/2$;
  else
    $\rho^{start} \leftarrow \rho$;
    $\rho \leftarrow$ 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

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 $\tau - 1$
$pro_S \leftarrow 1.0 - 0.5 * (\frac{count}{\tau-1})$;
return $(\hat{\epsilon}_i - losses) * pro_S$;

Budget Absorption: Budget Absorption uniformly allocates privacy budgets initially and remained privacy budgets can be absorbed by privacy budgets at the subsequent time-point. Given a database D^τ, to each row $D^\tau[i]$, at first it allocates $\hat{\epsilon}_i/w_i$, and then additionally allocates the previous remained budget $\hat{\epsilon}_i^{\tau-1} - \epsilon_i^{\tau-1}$. Thus, the privacy budget for $D^\tau[i]$ might be equal to $\hat{\epsilon}_i/w_i + \hat{\epsilon}_i^{\tau-1} - \epsilon_i^{\tau-1}$. However, if the total remained budget $\hat{\epsilon}_i - \sum_{t=\tau-w_i+1}^{\tau-1} \epsilon_i^t$ is less than $\hat{\epsilon}_i/w_i + \hat{\epsilon}_i^{\tau-1} - \epsilon_i^{\tau-1}$, it just allocates $\hat{\epsilon}_i - \sum_{t=\tau-w_i+1}^{\tau-1} \epsilon_i^t$ to $D^\tau[i]$.

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**
  $absorption \leftarrow \hat{\epsilon}_i^\tau + \hat{\epsilon}_i^{\tau-1} - \epsilon_i^{\tau-1}$
  $losses \leftarrow \sum_{t=\tau-w_i+1}^{\tau-1} \epsilon_i^t$;
  $remaining \leftarrow \hat{\epsilon}_i - losses$; //remaining privacy loss bound;
  $\hat{\epsilon}_i^\tau \leftarrow min(absorption, remaning)$;
end if
return $\hat{\epsilon}_i^\tau$;

The original 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 a 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 original BA is that it requires Laplace Mechanism as perturbation mechanism. Besides, the dissimilarity calculation sub-mechanism provides better utility over the whole timeline, but the utility might be worse for a single time-point because it also perturbs the former distance dis_1. 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: from data owner' point of view, to verify the effectiveness of budget allocation algorithms in terms of making full use of data owners' privacy loss bounds; from data buyers' point of view, to verify the effectiveness of two arbitrage-free trading protocols in terms of making more queries answerable and cost-performance.

Experiment setup: We generate n (the default value is 200) data owners' locations for 100 time-points. In this paper, because 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 [9], 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 $[W-1, W-1]$ where the default value of 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 data owners' privacy loss bounds. We use two budget allocation algorithms as benchmarks: *Timeline Uniform* [11] 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).

Request the same histogram variance v. First, we simulate the cases where the buyer requests the same histogram variance for all time-points, because query answers from different time-points but with the same utility can be easily used to observe the changes of the number of data owners in different regions. Figure 4 shows the average privacy loss consumed per time-point and data owner. For both the UniformTrading and PersonalizedTrading protocols, if the buyer requests a low histogram variance which corresponds to a high utility, the Seize-the-moment algorithm always outperforms the others because it can help sell more privacy losses for data owners on average. However, with the histogram variance increasing, the gap of performance between those algorithms narrows and disappears finally. Hence, if buyers always want accurate queries, the superiority of Seize-the-moment will be highlighted. Unsurprisingly, 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.

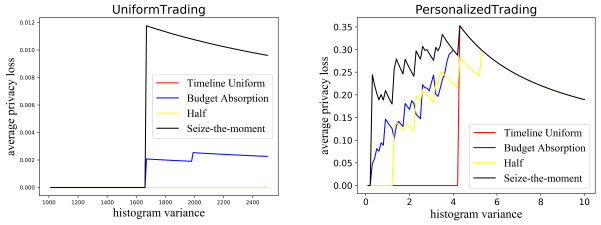

Fig. 4. Request the same histogram variance over the timeline

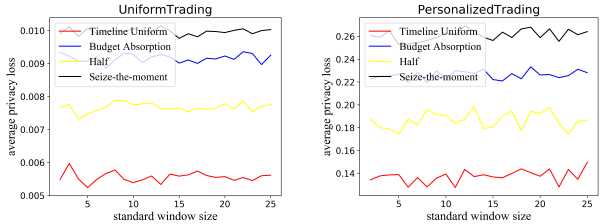

Fig. 5. Impact of sliding-window size on average privacy loss

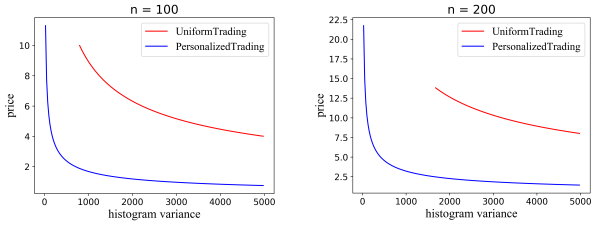

Fig. 6. Impact of the number of users

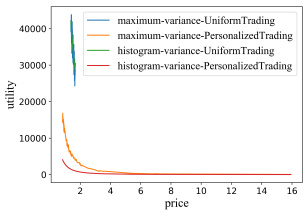

Fig. 7. Actual Utility

Varying the standard sliding-window size W. Then we want to know the impact of the W which controls the range of each data owner's preferred sliding-window size w_i. This time for each query the buyer randomly choose a histogram variance v more than the lower bound $\check{v}$, which means each query is answerable. As shown in Figure 5, it seems that the sliding-window size has slight impact on the average privacy loss consumed. However, Seize-the-moment still performs the best for all the standard sliding-window sizes for both UniformTrading and PersonalizedTrading.

B. Experiments for Data Buyers

Next, we evaluate PersonalizeTrading and UniformTrading, both with Seize-the-moment for *BudgetAlloc* due to its great performance, for our design goals of raising cost-performance and making more queries answerable for data buyers.

Impact of the number of data owners n.: As depicted in Figure 6, as the n increases, PersonalizedTrading can always make more queries answerable because its lower bound of histogram variance is much lower than UniformTrading; its cost-performance is also better because for the same histogram variance, the price is lower. Remind that Laplace Mechanism, the perturbation mechanism for UniformTrading, always use the minimum privacy loss to perturb a query answer, resulting in UniformTrading more sensitive to an extremely conservative data owner. Thus, when n increases, there will be more extremely conservative data owners involved in data trading, which causes the bad performance of UniformTrading in terms of making more queries answerable.

Actual utility. Since the histogram variance v is a worst-case guarantee which satisfies $v \geq \max_l Var[\tilde{q}_l^\tau]$, we measure the actual utility of a query answer by the maximum variance among all its elements, i.e., $v^{max} = \max_l Var[\tilde{q}_l^\tau]$. As shown in Figure 7, as for UniformTrading, there is no difference between the histogram variance and the maximum variance for the same query answer; but as for PersonalizedTrading, the maximum variance is obviously lower, which means that the accuracy of query answers we trade is actually higher than we guarantee. Thus, the cost-performance of PersonalizedTrading actually is even better than we observed in Figure 6.

VI. Related Work

Personal Data Trading: Balazinska et al. [14] guides the tendency of research on *Data Market* for the database research community. In recent years, personal data or individual privacy loss has been perceived as a commodity. Ghosh et al. [3] designed markets for trading statistics over private data and the privacy loss at auctions. Then, Riederer et al. [15] focused on allowing users to decide which part of their personally identifiable information for sale in the auction. Koutris et al. [5] proposed the original model of query-based pricing where no negotiation is allowed in terms of pricing. Li et al. [6] also adapted the query-based model with the constraint of arbitrage-free and privacy preservation by differential privacy, but individual privacy loss can be infinite. Nget et al. [9] analyzed people's privacy attitude and found that some people put privacy in a more important place than money. Hence in this paper, to bound individual privacy loss, we let each data owner set the upper bound of her own privacy loss. Niu et al. [7] proposed a pricing framework trading common aggregate statistics over correlated data. As for mobile data, a data-sharing mechanism and decision framework was proposed by Aly et al. to estimate the expected value of a single data point and make purchasing decisions [16]; Kanza et al. [17] also presented a geosocial marketplace taking privacy protection into account but without consideration of arbitrage-free pricing. We consider trading statistical queries over infinite trajectory data

streams rather than trading a raw trajectory data point, with the constraint of arbitrage-free pricing.

Differentially Private Streaming Data Release: The research on differentially private streaming data was initiated by the work of Dwork et al. [18]: event-level DP which protects at most one single event, and user-level DP which hides all the events of each user. Their another work [19] focused on finite data streams and a binary tree is constructed to inject an appropriate noise. Then, Chan et al. [20] adapted such technique for infinite streams. Bolot et al. [21] proposed the notion of *decayed privacy* to reduce the privacy protection level of previous data. Recently, Kellaris et al. [11] tackled the limitations of event-level DP and user-level DP, and introduced the notion of *w-event privacy* where a sliding window methodology is applied. They also proposed budget allocation algorithms budget distribution (BD) and budget absorption (BA) which allocate portion of the entire privacy budget for approximation of data publishing and portion for data perturbation. Wang et al. [22] [23] further proposed an adaptive budget allocation algorithm which dynamically computes the portion to increase the utility of the released data. Then, in order to personalize the fixed parameter w of window size, and the rate of data points being generated, Cao et al. [24] extended w-event privacy to l-trajectory privacy. Also, they investigated data correlations in streaming data which causes temporal privacy loss, and proposed mechanisms to bound such privacy loss [25] [26]. We combine w-event privacy and personalized differential privacy together to publish streaming data to make the level of privacy protection personalized [21] for each data owner.

VII. Conclusion

We proposed a trading framework for infinite streaming mobile data where each data owner's privacy loss under personalized $\mathcal{W}$-event privacy is bounded. We proposed Theorem 3.1 to guide the design of framework to guarantee arbitrage-free pricing for the market maker. Then, also based on this theorem, we designed the PersonalizedTrading protocol to make more queries answerable and raise the cost-performance for data buyers, and our experiments verified its outperforming effectiveness compared to the baseline UniformTrading protocol. The experimental results also show the importance of trading personal data with personalized privacy losses, since uniform privacy losses extremely constrain the utility of query answers so that a huge number of queries cannot be answered.

References

[1] J. Brustein, "Start-ups aim to help users put a price on their personal data," *The New York Times*, Feb 2012.

[2] J. A. Gabisch and G. R. Milne, "The impact of compensation on information ownership and privacy control," *Journal of Consumer Marketing*, vol. 31, no. 1, pp. 13–26, 2014.

[3] A. Ghosh and A. Roth, "Selling privacy at auction," *Games and Economic Behavior*, vol. 91, pp. 334–346, 2015.

[4] P. Koutris, P. Upadhyaya, M. Balazinska, B. Howe, and D. Suciu, "Toward practical query pricing with querymarket," in *proceedings of the 2013 ACM SIGMOD international conference on management of data*. ACM, 2013, pp. 613–624.

[5] ——, "Query-based data pricing," *Journal of the ACM (JACM)*, vol. 62, no. 5, p. 43, 2015.

[6] 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, p. 34, 2014.

[7] 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*. ACM, 2018, pp. 2031–2040.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] S. Zheng, Y. Cao, and M. Yoshikawa, "Money cannot buy everything: Trading mobile data with controllable privacy loss (full version)," https://www.db.soc.i.kyoto-u.ac.jp/mdm20.pdf.

[14] 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.

[15] C. Riederer, V. Erramilli, A. Chaintreau, B. Krishnamurthy, and P. Rodriguez, "For sale : your data: by : you," in *Acm Workshop on Hot Topics in Networks*, 2011.

[16] 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*, ser. SIGSPATIAL '18. New York, NY, USA: ACM, 2018, pp. 179–188. [Online]. Available: http://doi.acm.org/10.1145/3274895.3274905

[17] Y. Kanza and H. Samet, "An online marketplace for geosocial data," in *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 2015, p. 10.

[18] 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*. ACM, 2010, pp. 715–724.

[19] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[20] T.-H. H. Chan, E. Shi, and D. Song, "Private and continual release of statistics," *ACM Transactions on Information and System Security (TISSEC)*, vol. 14, no. 3, p. 26, 2011.

[21] J. Bolot, N. Fawaz, S. Muthukrishnan, A. Nikolov, and N. Taft, "Private decayed predicate sums on streams," in *Proceedings of the 16th International Conference on Database Theory*, ser. ICDT '13. New York, NY, USA: ACM, 2013, pp. 284–295.

[22] 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.

[23] ——, "Rescuedp: Real-time spatio-temporal crowd-sourced data publishing with differential privacy," in *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*. IEEE, 2016, pp. 1–9.

[24] 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.

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

[26] ——, "Quantifying differential privacy in continuous data release under temporal correlations," *IEEE Transactions on Knowledge and Data Engineering*, 2018.

APPENDIX

Proof A.1 (of Theorem 3.1): 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 are going to construct a function $h(x)$ ($x \geq 0$) which we will prove sub-additive later, i.e., $\forall x_1, x_2 \geq 0$, $h(x_1) + h(x_2) \geq h(x_1 + x_2)$.

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

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

$$
\begin{aligned}
&\forall \epsilon > 0, U(\rho \cdot \epsilon) \cdot U''(\rho \cdot \epsilon) - 2 \cdot [U'(\rho \cdot \epsilon)]^2 \leq 0 \\
\Rightarrow &\forall \epsilon > 0, U_\rho(\epsilon) \cdot U_\rho''(\epsilon) - 2 \cdot [U_\rho'(\epsilon)]^2 \leq 0 \\
\Rightarrow &\forall v > 0, v \cdot [-\frac{U_\rho^{-1\prime\prime}(v)}{(U_\rho^{-1\prime}(v))^3}] - 2 \cdot (\frac{1}{U_\rho^{-1\prime}(v)})^2 \leq 0 \\
\Rightarrow &\forall v > 0, v \cdot U_\rho^{-1\prime\prime}(v) + 2 \cdot U_\rho^{-1\prime}(v) \leq 0 \\
\Rightarrow &\forall x > 0, \frac{1}{x} \cdot U_\rho^{-1\prime\prime}(\frac{1}{x}) + 2 \cdot U_\rho^{-1\prime}(\frac{1}{x}) \leq 0 \\
\Rightarrow &\forall x > 0, \frac{1}{x^4} \cdot U_\rho^{-1\prime\prime}(\frac{1}{x}) + \frac{2}{x^3} \cdot U_\rho^{-1\prime}(\frac{1}{x}) \leq 0 \\
\Rightarrow &\forall x > 0, h''(x) = U_\rho^{-1\prime\prime}(\frac{1}{x}) \cdot [(\frac{1}{x})']^2 + U_\rho^{-1\prime}(\frac{1}{x}) \cdot (\frac{1}{x})'' \leq 0
\end{aligned}
$$

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 $U_\rho^{-1}(v) \leq \sum_{j=1}^m U_\rho^{-1}(v_j)$:

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

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

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

Proof A.2 (of Theorem 3.2): For any pair of w_i-neighboring stream prefixes S^τ, $S^{\tau\prime}$ with pairs of neighboring databases $D^t \overset{u_i}{\sim} D^{t\prime}$, any τ and any possible output $o = (o_1, ..., o_\tau)$, Due to Definition 2.1, we have:

$$
\forall t \in [\tau], Pr[\mathcal{M}_t(D^t) = o_t] \leq e^{\epsilon_i^t} Pr[\mathcal{M}_t(D^{t\prime}) = o_t]
$$

Because each $\mathcal{M}_t$ computes o_t independently, we have:

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

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

$$
\begin{aligned}
\frac{Pr[\mathcal{M}(S^\tau) = o]}{Pr[\mathcal{M}(S^{\tau\prime}) = o]} &= \frac{\prod_{k=t-w_i+1}^{t} Pr[\mathcal{M}_k(D^k) = o_k]}{\prod_{k=t-w_i+1}^{t} Pr[\mathcal{M}_k(D^{k\prime}) = o_k]} \\
&\leq \prod_{k=t-w_i+1}^{t} e^{\epsilon_i^k} = e^{\sum_{k=t-w_i+1}^{t} \epsilon_i^k} \leq e^{\epsilon_i}
\end{aligned}
$$

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