A Theory of Pricing Private Data

CHAO LI, University of Massachusetts, Amherst
DANIEL YANG LI, University of Washington, Seattle
GEROME MIKLAU, University of Massachusetts, Amherst
DAN SUCIU, University of Washington, Seattle

Personal data has value to both its owner and to institutions who would like to analyze it. Privacy mechanisms protect the owner's data while releasing to analysts noisy versions of aggregate query results. But such strict protections of the individual's data have not yet found wide use in practice. Instead, Internet companies, for example, commonly provide free services in return for valuable sensitive information from users, which they exploit and sometimes sell to third parties.

As awareness of the value of personal data increases, so has the drive to compensate the end-user for her private information. The idea of monetizing private data can improve over the narrower view of hiding private data, since it empowers individuals to control their data through financial means.

In this article we propose a theoretical framework for assigning prices to noisy query answers as a function of their accuracy, and for dividing the price amongst data owners who deserve compensation for their loss of privacy. Our framework adopts and extends key principles from both differential privacy and query pricing in data markets. We identify essential properties of the pricing function and micropayments, and characterize valid solutions.

Categories and Subject Descriptors: H.2.8 [**Database Applications**]: Statistical databases

General Terms: Theory, Economics

Additional Key Words and Phrases: Differential privacy, data pricing, arbitrage

ACM Reference Format:
Chao Li, Daniel Yang Li, Gerome Miklau, and Dan Suciu. 2014. A theory of pricing private data. ACM Trans. Datab. Syst. 39, 4, Article 34 (December 2014), 28 pages.
DOI: http://dx.doi.org/10.1145/2691190.2691191

1. INTRODUCTION

Personal data has value to both its owner and to institutions who would like to analyze it. The interests of individuals and institutions with respect to personal data are often at odds and a rich literature on privacy-preserving data publishing techniques [Fung et al. 2010; Chen et al. 2010] has tried to devise technical methods for negotiating these competing interests. Broadly construed, privacy refers to an individual's right to control how her private data will be used, and was originally phrased as an individual's right to be protected against gossip and slander [Danezis and Gürses 2010]. Research on privacy-preserving data publishing has focused more narrowly on privacy as data

C. Li was supported by NSF CNS-1012748, G. Miklau was partially supported by NSF CNS-1012748, NSF CNS-0964094, and the European Research Council under the Webdam grant, and D. Li and D. Suciu were supported by NSF IIS-0915054 and NSF CCF-1047815.
Authors' addresses: C. Li, School of Computer Science, University of Massachusetts, Amherst, MA; D. Y. Li, Department of Computer Science and Engineering, University of Washington, Seattle, WA; G. Miklau, School of Computer Science, University of Massachusetts, Amherst, MA; D. Suciu (corresponding author), Department of Computer Science and Engineering, University of Washington, Seattle, WA; email: suciu@cs.washington.edu.

DOI: http://dx.doi.org/10.1145/2691190.2691191

confidentiality. For example, in perturbation-based data privacy, the goal is to protect an individual's personal data while releasing to legitimate users the result of aggregate computations over a large population [Dwork 2011].

To date, this goal has remained elusive. One important result from this line of work is that any mechanism providing reasonable privacy must strictly limit the number of query answers that can be accurately released [Dinur and Nissim 2003], thus imposing a strict *privacy budget* for any legitimate user of the data [McSherry 2010]. Researchers are actively investigating formal notions of privacy and their implications for effective data analysis. Yet, with rare exception [Kifer et al. 2008], perturbation-based privacy mechanisms have not been deployed in practice.

Instead, many Internet companies have followed a simple formula to acquire personal data. They offer a free service, attract users who provide their data, and then monetize the personal data by selling it, or information derived from it, to third parties. A recent study by JPMorgan Chase [Brustein 2012] found that each unique user is worth approximately $4 to Facebook and $24 to Google.

Currently, many users are willing to provide their private data in return for access to online services. But as individuals become more aware of the use of their data by corporate entities, of the potential consequences of disclosure, and of the ultimate value of their personal data, there has been a drive to compensate them directly [World Economid Forum 2011]. In fact, startup companies are currently developing infrastructure to support this trend. For example, www.personal.com creates personal data vaults, each of which may contain thousands of data points about its users. Businesses pay for this data, and the data owners are appropriately compensated.

Monetizing private data is an improvement over the narrow view of privacy as data confidentiality because it empowers individuals to control their data through financial means. In this article we propose a framework for assigning prices to queries in order to compensate the data owners for their loss of privacy. Our framework borrows from, and extends, key principles from both differential privacy [Dwork 2011] and data markets [Koutris et al. 2012; Li and Miklau 2012]; a preliminary abstract of this article appeared in Li et al. [2013]. There are three actors in our setting: individuals, or data *owners*, contribute their personal data; a *buyer* submits an aggregate query over many owners' data; and a *market maker*, trusted to answer queries on behalf of owners, charges the buyer and compensates the owners. Our framework makes three important connections.

Perturbation and Price. In response to a buyer's query, the market maker computes the true query answer, adds random noise, and returns a perturbed result. While under differential privacy perturbation is always necessary, here query answers could be sold unperturbed, but the price would be high because each data owner contributing to an aggregate query needs to be compensated. By adding perturbation to the query answer, the price can be lowered: the more perturbation, the lower the price. The buyer specifies how much accuracy he is willing to pay for when issuing the query. Unperturbed query answers are very expensive, but at the other extreme, query answers are almost free if the noise added is the same as in differential privacy [Dwork 2011] with conservative privacy parameters. The relationship between the accuracy of a query result and its cost depends on the query and the preferences of contributing data owners. Formalizing this relationship is one of the goals of this work.

Arbitrage and Perturbation. Arbitrage is an undesirable property of a set of priced queries that allows a buyer to obtain the answer to a query more cheaply than its advertised price by deriving the answer from a less expensive alternative set of queries. As a simple example, suppose that a given query is sold with two options for perturbation, measured by variance: a variance of 10 for $5 and a variance of 1

for $200. A savvy buyer who seeks a variance of 1 would never pay $200. Instead, he would purchase the first query 10 times, receive 10 noisy answers, and compute their average. Since the noise is added independently, the variance of the resulting average is 1, and the total cost is only $50. Arbitrage opportunities result from inconsistencies in the pricing of queries which must be avoided, and perturbing query answers makes this significantly more challenging. Avoiding arbitrage in data markets has been considered before only in the absence of perturbation [Balazinska et al. 2011; Koutris et al. 2012; Li and Miklau 2012]. Formalizing arbitrage for noisy queries is a second goal of this article. While, in theory, achieving arbitrage freeness requires imposing a lower bound on the ratio between the price of low-accuracy and high-accuracy queries, we will show it is possible to design quite flexible arbitrage-free pricing functions.

Privacy loss and Payments. Given a randomized mechanism for answering a query q, a common measure of privacy loss to an individual is defined by differential privacy: it is the maximum ratio between the probability of returning some fixed output with and without that individual's data. Differential privacy imposes a bound of e^{ε} on this quantity, where ε is a small constant, presumed acceptable to all individuals in the population. Our framework contrasts with this in several ways. First, the privacy loss is not limited a priori, but depends on the buyer's request. If the buyer asks for a query with low variance, then the privacy loss to individuals will be high. These data owners must be compensated for their privacy loss through the buyer's payment. At an extreme, if the query answer is exact (unperturbed), then the privacy loss to some individuals could be total and they must be compensated appropriately. Also, we allow each data owner to value his privacy loss separately, by demanding greater or lesser payments. Formalizing the relationship between privacy loss and payments to the data owners is a third goal of this article.

By charging buyers for access to private data, we overcome a fundamental limitation of perturbation-based privacy-preserving mechanisms, namely the privacy budget. This term refers to a limit on the quantity and/or accuracy of queries that any buyer can ask in order to prevent an unacceptable disclosure of the data. For example, if a differentially private mechanism adds Laplacian noise with variance v then, by asking the same query n times, the buyer can reduce the variance to v/n. Even if queries are restricted to aggregate queries, there exist sequences of queries that can reveal the private data for most individuals in the database [Dinur and Nissim 2003] and enforcing the privacy budget must prevent this. In contrast, when private data is priced, full disclosure is possible only if the buyer pays a high price. For example, in order to reduce the variance to v/n, the buyer would have to purchase the query n times, thus paying n times more than for a single query. In order to perform the attacks in Dwork and Yekhanin [2008], he would have to pay for (roughly) n queries.

Thus, the burden of the market maker is no longer to guard the privacy budget, but instead to ensure that prices are set such that, whatever disclosure is obtained by the buyer, all contributing individuals are properly compensated. In particular, if a sequence of queries can indeed reveal the private data for most individuals, its price must approach the total cost for the entire database.

The article is organized as follows. We describe the basic framework for pricing private data in Section 2. In Section 3, we discuss the main required properties for pricing functions, developing notions of answerability for perturbed query answers and characterizing arbitrage-free pricing functions. At the end of that section, in Section 3.7, we discuss key assumptions about query answerability and adversary capabilities that we have made in this work. In Section 4 we develop a notion of personalized privacy loss for individuals based on differential privacy.

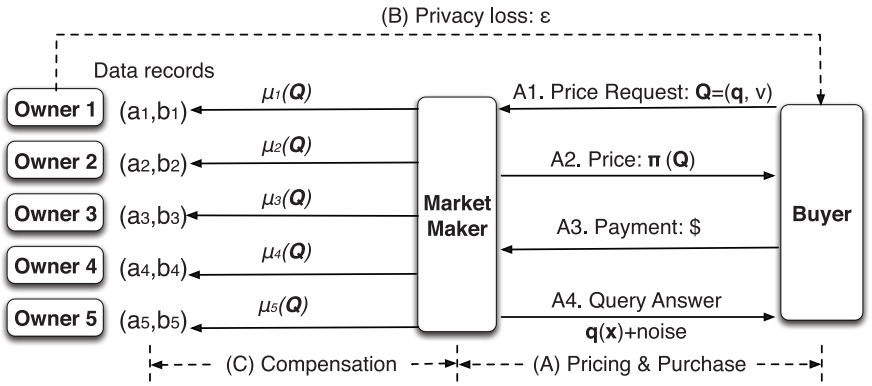

Fig. 1. The pricing framework has three components: (A) pricing and purchase, where the buyer asks a query $\mathbf{Q} = (\mathbf{q}, v)$ and must pay its price $\pi(\mathbf{Q})$; (B) privacy loss where, by answering $\mathbf{Q}$, the market maker leaks some information ε about the private data of the data owners to the buyer; (C) compensation, where the market maker must compensate each data owner for her privacy loss with micropayments; $\mu_i(\mathbf{Q})$ is the total amount of micropayments for all users in bucket i. The pricing framework is *balanced* if the price $\pi(\mathbf{Q})$ is sufficient to cover all micropayments μ_i and the micropayments μ_i compensate the owners for their privacy loss ε.

In Section 5 we first define micropayment functions using this measure of privacy loss and then define balanced frameworks, bringing together query prices, privacy loss, and micropayments. We discuss three future challenges for pricing private data in Section 6: disclosures that could result from an individual's privacy valuations alone, how to set the price for "asking price", and incentives for data owners to honestly reveal the valuations of their data. We discuss related work and conclude in Section 7 and Section 8.

2. BASIC CONCEPTS

In this section we describe the basic architecture of the private data pricing framework as illustrated in Figure 1.

2.1. The Main Actors

We describe in the following text the main actors in our proposed marketplace: data owners who contribute data, query buyers, and a market maker negotiating between the two.

The Market Maker. The market maker is trusted by the buyer and by each of the data owners. He collects data from the owners and sells it in the form of queries. When a buyer decides to purchase a query, the market maker collects payment, computes the answer to the query, adds noise as appropriate, returns the result to the buyer, and finally distributes individual payments to the data owners. The market maker may retain a fraction of the price as profit.

The Owner and Her Data. Each owner contributes a single tuple conforming to a relational schema $R(\mathbb{A})$, with attributes $\mathbb{A} = \{A_1, A_2, \ldots, A_k\}$. The crossproduct of the attribute domains, written $dom(\mathbb{A})$, is the set of all possible tuples that could occur. For a fixed k, its size $n = |dom(\mathbb{A})|$ is polynomial in the number of users; n grows exponentially with k. Having collected tuples from each owner, the market maker forms a relational table I, an instance of $R(\mathbb{A})$.

The Buyer and His Queries. The buyer is a data analyst who wishes to compute some queries over the data. We restrict our attention to linear aggregation queries

over instance I, a common class of queries that has received considerable attention from the privacy community [Li et al. 2010; Yuan et al. 2012; Hardt et al. 2012].

To express linear queries, we first represent the instance I by a finite *data vector* $\mathbf{x}$ consisting of nonnegative integral counts. For each element $t \in dom(\mathbb{A})$, the vector $\mathbf{x}$ has one entry that reports the number of individuals whose tuple matches t. We assume that $dom(\mathbb{A})$ is ordered, and denote x_i the i'th entry in the vector $\mathbf{x}$. In other words, x_i represents the number of individuals whose attribute values match the i'th entry in $dom(\mathbb{A})$. Notice that, although the size n of the vector $\mathbf{x}$ can be large, since it is exponential in the number of attributes, the vector is sparse, having nonzero counts only for tuples present in I. In practice one avoids fully materializing $\mathbf{x}$, and retains only the relational representation of I.

Definition 2.1 (*Linear Query*). A linear query is a real-valued vector $\mathbf{q} = (q_1, q_2 \ldots q_n)$. The answer $\mathbf{q}(\mathbf{x})$ to a linear query on $\mathbf{x}$ is the vector product $\mathbf{qx} = q_1x_1 + \cdots + q_nx_n$.

A linear query with only coefficients of zero or 1 can express any predicate counting query, counting the number of individuals satisfying any boolean predicate over the attributes in the schema. This includes many common statistical queries such as histogram counts, data cube queries, and marginals. With more general coefficients, linear queries can express differences, weighted counts, and averages over discrete domains. While the query vector $\mathbf{q}$ is large, in practice queries are expressed compactly, directly over the relational representation of the data, in a query language like SQL.

Example 2.2. Consider a competition between candidates A and B that is decided by a population of voters. The ratings and descriptive fields of voters are sensitive and should be compensated properly if used in any way. Imagine that users contribute their gender and age along with two numerical ratings, each consisting of values from the domain $\{0, 1, 2, 3, 4, 5\}$. Thus Alice's tuple may be ('Female', 39, 0, 5) if she strongly favors candidate B over candidate A. If the domain for the age attribute is [1..120], then the database vector will have size $2 \times 120 \times 6 \times 6 = 8640$.

Examples of linear queries include the total of all ratings for candidate A, the number of female voters over 40 who gave candidate A a rating of 5, or the number of voters whose rating for candidate A exceeds that for candidate B.

We make two comments about our framework. First, although the vector $\mathbf{x}$ removes personal identifiable information, since x_i does not represent an individual user but rather the total number of users with a particular combination of attribute values, queries can still leak information about individual users. Continuing our example, a buyer may know that Alice is 39 years old. Then, he could issue 36 linear queries, requesting the counts for all entries of the form ('Female', 39, a, b) for all combinations of values a, b: if the database has only a small number of female users of age 39, then most of the 36 answers are zero and the buyer has a pretty good guess of Alice's possible votes. Research on differential privacy is concerned with preventing such leaks.

Second, we assume the buyer is allowed to issue multiple queries, which means that he can combine information derived from multiple queries to infer answers to other queries not explicitly requested. This presents a challenge we must address: to ensure that the buyer pays for any information that he might derive, whether directly or indirectly.

2.2. Balanced Pricing Framework

The market maker enters two contracts: (1) It promises to answer the buyer's queries according to an agreed price π, (Section 3) and (2) it promises to compensate the data owners with a micropayments $\mu_i(\varepsilon)$ whenever they suffer a privacy loss ε in response to a buyer's query (Section 5).

In the contract with the buyer, the market maker allows the buyer to specify, for each linear query $\mathbf{q}$, an amount of noise v that he is willing to tolerate in the answer. Adding noise reduces the price. Thus, the buyer's query is a pair $\mathbf{Q} = (\mathbf{q}, v)$, where $\mathbf{q}$ is a linear query and $v \geq 0$ represents an upper bound on the variance, and the price depends on both, $\pi(\mathbf{Q}) = \pi(\mathbf{q}, v) \geq 0$. The market maker answers by first computing the exact answer $\mathbf{q}(\mathbf{x})$, then adding noise sampled from a distribution with mean 0 and variance at most v. This feature gives the buyer more pricing options because, by increasing v, he can lower his price. Notice that the price depends only on the variance v, and not on the type of noise.

In the contract with the data owner, the market maker promises to compensate her with a micropayment. We denote $\mu_i(\varepsilon)$ the sum of all micropayments to all users whose attributes match the i'th entry to the vector, where ε is the privacy loss incurred by answering the query $\mathbf{Q}$. The privacy loss depends both on the variance and on the type of noise that the market maker uses to answer queries: in Section 4 we will restrict the noise to the Laplace distribution, for which there exists an explicit formula connecting the privacy loss ε to the variance. In this case the micropayment depends on the buyer's query $\mu_i(\mathbf{Q})$. For the pricing framework to be balanced, we must have $\sum_{i=1}^{n} \mu_i(\mathbf{Q}) \leq \pi(\mathbf{Q})$. Note that $\mu_i(\mathbf{Q})$ needs to be further split among all users participating in the i'th bucket of the data $\mathbf{x}$.

Example 2.3. Continuing Example 2.2, suppose there are 1000 voters and that Bob, the buyer, wants to compute the sum of ratings for candidate A. Assume that each voter requires \$10 for each raw vote. For an accurate answer to the query, Bob needs to pay \$10000, which is, arguably, too expensive.

Assume Bob is willing to buy the query perturbed with variance $v = 5{,}000$, which gives an error[1] of ± 300 with 94% confidence. The market maker should charge Bob a much lower price; to see how low, we need to consider how the market maker compensates the data owners. We assume he uses Laplacian noise for the perturbation, and therefore the answer to the query is ε-differentially private[2], with $\varepsilon = 0.1$, which offers reasonably good privacy to all data owners; each will be happy to accept only \$0.001 for basically no loss of privacy, and Bob pays only \$1 for the entire query. The challenge is to design the prices *in between*. For example, suppose the data owner wants to buy more accuracy, say a variance $v = 50$ (to reduce the error to ± 30), what should the price be now? We will answer this in Example 3.17. For now, let us observe that the price cannot exceed \$100. If it did, then a savvy buyer would never pay that price, instead purchasing the \$1 query 100 times, computing the average, and obtaining the answer with a variance of $5000/100 = 50$. This is an example of arbitrage and the market maker should define a pricing function that avoids it.

While in the contract with the buyer the price depends only on the variance and not on the type of perturbation, the contract with the data owner is highly sensitive to the type of noise. For example, consider this noise $P(0) = 1 - 2/m$, $P(\pm m) = 1/m$, where $m = 10^{64}$ (mean 0, variance $2m$) is a poor choice. On one hand, it has a high variance, which implies a low price π. On the other hand, it returns an accurate answer with extremely high probability, leading to huge privacy losses ε_i and consequently to huge micropayments. The market maker will not be able to recover his costs; thus, in order to design a balanced pricing framework, we need to have a perturbation mechanism where the privacy loss is given by an explicit function in the variance.

[1] $\Pr(|\hat{q} - q| \geq 3\sqrt{2} \cdot \sigma) \leq 1/18 = 0.056$ (Chebyshev's inequality), where $\sigma = \sqrt{v} = 50\sqrt{2}$. Thus, the error is $\leq 3\sqrt{2} \cdot \sigma = 300$ with probability $\leq 5.6\%$.

[2] $\varepsilon = \sqrt{2} \cdot \text{sensitivity}(\mathbf{q})/\sigma = 5\sqrt{2}/50\sqrt{2} = 0.1$.

In Section 5.1 we will only consider the Laplacian mechanism, where such a function exists.

3. PRICING QUERIES

In this section we describe the first component of the framework in Figure 1: the pricing function $\pi(\mathbf{Q}) = \pi(\mathbf{q}, v)$. We denote $\mathbb{R}^+ = [0, \infty)$ and $\bar{\mathbb{R}}^+ = \mathbb{R}^+ \cup \{\infty\}$.

Definition 3.1. A pricing function is $\pi : \mathbb{R}^n \times \bar{\mathbb{R}}^+ \to \bar{\mathbb{R}}^+$.

In our framework, the buyer is allowed to issue multiple queries. As a consequence, an important concern is that the buyer may combine answers from multiple queries and derive an answer to a new query without paying the full price for the latter, a situation we call *arbitrage*. A reasonable pricing function must guarantee that no arbitrage is possible, in which case we call it *arbitrage free*. Such a pricing function ensures that the market maker receives proper payment for each query by removing any incentive for the buyer to "game" the system by asking a set of cheaper queries in order to obtain the desired answer. In this section we formally define arbitrage-free pricing functions, study their properties, and discuss how to construct arbitrage-free pricing functions.

3.1. Queries and Answers

A *randomized mechanism* means a random function $\mathcal{K}$, with some input $\mathbf{x}$, denoted $\mathcal{K}(\mathbf{x})$. For a given query $\mathbf{Q} = (\mathbf{q}, v)$, the market maker answers it using a randomized mechanism $\mathcal{K}_\mathbf{Q}$, with the property that, for any $\mathbf{x}$, $\mathbf{E}\left(\mathcal{K}_\mathbf{Q}(\mathbf{x})\right) = \mathbf{q}(\mathbf{x})$ and $\mathbf{Var}\left(\mathcal{K}_\mathbf{Q}(\mathbf{x})\right) \leq v$. In other words, when the buyer asks for a query $\mathbf{Q}$, the market maker samples one value from the distribution $\mathcal{K}_\mathbf{Q}$ and returns it to the buyer in exchange for payment $\pi(\mathbf{Q})$. We abbreviate $\mathcal{K}_\mathbf{Q}$ with $\mathcal{K}$ when $\mathbf{Q}$ is clear from the context.

Definition 3.2. We say that a random function $\mathcal{K}(\mathbf{x})$ *answers the query* $\mathbf{Q} = (\mathbf{q}, v)$ *on the database* $\mathbf{x}$ if its expectation is $\mathbf{q}(\mathbf{x})$ and its variance less than or equal to v.

Other options for answering queries are possible and we briefly discuss them in Section 3.7.

To illustrate with a simple example, consider the mechanism $\mathcal{K}_\mathbf{Q}$ which, on input $\mathbf{x}$ first computes the query $\mathbf{q}(\mathbf{x})$, then adds a random noise with mean 0 and variance v. In this section we do not impose any restrictions on the type of perturbation used in answering the query. The contract between the buyer and the market maker refers only to the variance: the buyer pays for a certain variance, and the market maker must answer with at most this variance. The inherent assumption is that the buyer only cares about the variance and is indifferent to other properties of the perturbation. However, the market maker also has a contract with the data providers and needs to compensate them according to their privacy loss. Different mechanisms with the same variance will lead to different privacy losses; in Section 4 we will restrict the perturbation to consist of a Laplacian noise that offers a bound on the privacy loss that depends only on the variance.

We assume the market maker is stateless: he does not keep a log of previous users, their queries, or of released answers. As a consequence, each query is answered using an independent random variable. If the same buyer issues the same query repeatedly, the market maker answers using independent samples from the random function $\mathcal{K}$. Of course, the buyer would have to pay for each query separately.

3.2. Answerability and Determinacy

Before investigating arbitrage, we establish the key concept of query answerability. This notion is well studied for deterministic queries and views [Halevy 2001; Nash et al. 2010] but, in our setting, the query answers are random variables, and it requires

a precise definition. Our definition given next directly extends the traditional definition from deterministic to randomized queries.

Definition 3.3 (*Answerability*). A query $\mathbf{Q}$ is *answerable* from a multiset of queries $\mathbf{S} = \{\mathbf{Q_1}, \ldots, \mathbf{Q_k}\}$ if there exists a function $f : \mathbb{R}^k \to \mathbb{R}$ such that, for any mechanisms $\mathcal{K}_1, \ldots, \mathcal{K}_k$ that answer the queries $\mathbf{Q_1}, \ldots, \mathbf{Q_k}$, the composite mechanism defined as $\mathcal{K}(\mathbf{x}) \stackrel{\text{def}}{=} f(\mathcal{K}_1(\mathbf{x}), \ldots, \mathcal{K}_k(\mathbf{x}))$ answers the query $\mathbf{Q}$.

We say that $\mathbf{Q}$ is *linearly answerable* from $\mathbf{Q}_1, \ldots, \mathbf{Q}_k$ if the function f is linear.

In other words, we want to compute $\mathbf{q}(\mathbf{x})$ with variance v, and have access to some other query results (estimates) $y_1 = \mathcal{K}_1, \ldots, y_k = \mathcal{K}_k$. We simply apply a function f to these results, and return $y = f(y_1, \ldots, y_k)$; this result must be an unbiased estimate of $\mathbf{q}(\mathbf{x})$ and must have variance $\leq v$. For a simple example, consider queries $\mathbf{Q_1} = (\mathbf{q_1}, v_1)$ and $\mathbf{Q_2} = (\mathbf{q_2}, v_2)$ and mechanisms $\mathcal{K}_1$ and $\mathcal{K}_2$ that answer them. The query $\mathbf{Q_3} = ((\mathbf{q_1} + \mathbf{q_2})/2, (v_1 + v_2)/4)$ is answerable from $\mathbf{Q_1}$ and $\mathbf{Q_2}$ because we can simply sum and scale the answers returned by the two mechanisms, and $\mathbf{E}\left((\mathcal{K}_1 + \mathcal{K}_2)/2\right) = (\mathbf{E}(\mathcal{K}_1) + \mathbf{E}(\mathcal{K}_2))/2$, and $\mathbf{Var}\left((\mathcal{K}_1 + \mathcal{K}_2)/2\right) = (\mathbf{Var}(\mathcal{K}_1) + \mathbf{Var}(\mathcal{K}_2))/4$. Since the function is linear, we say the query is linearly answerable.

When $k = 1$ then, with some abuse of notation, we denote the singleton set $\mathbf{S} = \{\mathbf{Q}_1\}$ as $\mathbf{Q}_1$. When $k = 0$, then $\mathbf{Q} = (\mathbf{q}, v)$ is answerable from $\mathbf{S}$ iff $\mathbf{q}$ is a constant, and this happens iff $\mathbf{q} = 0$ (because $\mathbf{q}$ is a linear query).

How do we check whether a query can be answered from a given set of queries? In this article we give a partial answer by characterizing when a query is *linearly* answerable.

Definition 3.4 (*Determinacy*). The *determinacy relation* is a relation between a query $\mathbf{Q}$ and a multiset of queries $\mathbf{S} = \{\mathbf{Q}_1, \ldots, \mathbf{Q}_k\}$, denoted $\mathbf{S} \to \mathbf{Q}$, and defined by the following rules:

Summation. $\{(\mathbf{q}_1, v_1), \ldots, (\mathbf{q}_k, v_k)\} \to (\mathbf{q}_1 + \cdots + \mathbf{q}_k, v_1 + \cdots + v_k)$;
Scalar multiplication. $\forall c \in \mathbb{R}, (\mathbf{q}, v) \to (c\mathbf{q}, c^2 v)$;
Relaxation. $(\mathbf{q}, v) \to (\mathbf{q}, v')$, where $v \leq v'$; and
Transitivity. If $\mathbf{S}_1 \to \mathbf{Q}_1, \ldots, \mathbf{S}_k \to \mathbf{Q}_k$ and $\{\mathbf{Q}_1, \ldots, \mathbf{Q}_k\} \to \mathbf{Q}$, then $\bigcup_{i=1}^k \mathbf{S}_k \to \mathbf{Q}$.

The following proposition gives a characterization of linear answerability.

PROPOSITION 3.5. *Let* $\mathbf{S} = \{(\mathbf{q}_1, v_1), \ldots, (\mathbf{q}_m, v_m)\}$ *be a multiset of queries, and* $\mathbf{Q} = (\mathbf{q}, v)$ *a query. Then the following conditions are equivalent.*

(1) $\mathbf{Q}$ *is linearly answerable from* $\mathbf{S}$.
(2) $\mathbf{S} \to \mathbf{Q}$.
(3) There exists $c_1, \ldots, c_m$ *such that* $c_1\mathbf{q}_1 + \cdots + c_m\mathbf{q}_m = \mathbf{q}$ *and* $c_1^2 v_1 + \cdots + c_m^2 v_m \leq v$.

PROOF. (1 $\Leftrightarrow$ 3): Follows from the definition of linear answerability.
(2 $\Rightarrow$ 3): It is clear that, in the rules of the determinacy relation, summation, scalar multiplication, and relaxation are special cases of 3. For the transitivity rule, for each $i = 1, \ldots, k$, let f_i be a linear function such that $f_i(\mathbf{S}_i) = \mathbf{q}_i$ with variance no more than v_i and let f be a linear function such that $f(\mathbf{q}_1, \ldots, \mathbf{q}_k) = \mathbf{q}$ with variance no more than v. Then $f_0 = f(f_1(\mathbf{S}_1), \ldots, f_k(\mathbf{S}_k))$ is a linear function of $\bigcup_{i=1}^k \mathbf{S}_k$ and the variance introduced is no more than v.
(3 $\Rightarrow$ 2): Since

$$(\mathbf{q}_i, v_i) \to \left(c_i\mathbf{q}_i, c_i^2 v_i\right),$$

$$\left\{\left(c_1\mathbf{q}_1, c_1^2 v_1\right), \ldots, \left(c_m\mathbf{q}_m, c_m^2 v_m\right)\right\} \to \left(c_1\mathbf{q}_1 + \cdots + c_m\mathbf{q}_m, c_1^2 v_1 + \cdots + c_m^2 v_m\right) = \left(\mathbf{q}, c_1^2 v_1 + \cdots + c_m^2 v_m\right),$$

and

$$\left(\mathbf{q}, c_1^2 v_1 + \cdots + c_m^2 v_m\right) \to (\mathbf{q}, v),$$

we obtain $\mathbf{S} \to \mathbf{Q}$. □

Thus, determinacy fully characterizes linear answerability.

In this article we restrict our discussion to linear answerability; in other words, we assume the buyer will attempt to derive new answers from existing queries only by computing linear combinations. This is a limitation, and we discuss some of its implications (both good and bad) in Section 3.7. Thus, in the rest of the article we will use interchangeably the determinacy relation $\mathbf{S} \to \mathbf{Q}$ instead of linear answerability.

Deciding determinacy $\mathbf{S} \to \mathbf{Q}$ can be done in polynomial time using a quadratic program. The program first determines whether $\mathbf{q}$ can be represented as a linear combination of queries in $\mathbf{S}$. If the answer is yes, the quadratic program further checks whether there is a linear combination such that the variance of answering $\mathbf{q}$ is at most v.

PROPOSITION 3.6. *Verifying whether a multiset* $\mathbf{S}$ *of* m *queries determines a query* $\mathbf{Q}$ *can be done in PTIME*(m, n).

PROOF. Given a multiset $\mathbf{S} = \{(\mathbf{q}_1, v_1), \ldots, (\mathbf{q}_m, v_m)\}$ and a query $(\mathbf{q}, v)$, the following quadratic program outputs the minimum possible variance to answer $\mathbf{q}$ using linear combinations of queries in $\mathbf{S}$.

$$\begin{aligned} &\text{Given: } \mathbf{q}, \mathbf{q}_1, \ldots, \mathbf{q}_m, v_1, \ldots, v_m, \\ &\text{Minimize: } c_1^2 v_1 + \cdots + c_m^2 v_m, \\ &\text{Subject to: } c_1 \mathbf{q}_1 + \cdots + c_m \mathbf{q}_m = \mathbf{q}. \end{aligned} \tag{1}$$

Once the quadratic program is solved, one can compare $c_1^2 v_1 + \cdots + c_m^2 v_m$ with v. According to Proposition 3.5, $\mathbf{S} \to (\mathbf{q}, v)$ if and only if $c_1^2 v_1 + \cdots + c_m^2 v_m \leq v$. Since the quadratic program given earlier has m variables and the constraints are a linear equation on n-dimensional vectors, it can be solved in PTIME(m, n) [Boyd and Vandenberghe 2004]. Thus the verification process can be done in PTIME(m, n) as well. □

Our main use of determinacy is in the definition of arbitrage (which we will introduce next): an adversary uses determinacy to attempt to answer the query $(\mathbf{q}, v)$ at a lower price. We note that PTIME(m, n) is not necessarily practical for the adversary, because the number of constraints in the quadratic program, Eq. (1), is n, which is exponential in the number of attributes (Section 2). If the relational table I were known, then one could reduce the number of constraints to $|I|$, since we only need one constraint in Eq. (1) for each combination of attribute values in I, but, of course, I is unknown to an adversary. Nevertheless, we will assume an adversary is powerful enough to check determinacy, and will design pricing functions that prevent arbitrage.

3.3. Arbitrage-Free Pricing Functions: Definition

Arbitrage is possible when the answer to a query $\mathbf{Q}$ can be obtained more cheaply than the advertised price $\pi(\mathbf{Q})$ from an alternative set of priced queries. When arbitrage is possible, it complicates the interface between the buyer and market maker: the buyer may need to reason carefully about his queries to achieve the lowest price, while at the same time the market maker may not achieve the revenue intended by some of his advertised prices. Thus arbitrage is undesirable, and we want to design pricing functions that are arbitrage free.

Definition 3.7 (*Arbitrage Free*). A pricing function $\pi(\mathbf{Q})$ is arbitrage free if, for every multiset $\mathbf{S} = \{\mathbf{Q}_1, \ldots, \mathbf{Q}_m\}$, if $\mathbf{S} \to \mathbf{Q}$ then

$$\pi(\mathbf{Q}) \leq \sum_{i=1}^{m} \pi(\mathbf{Q}_i).$$

The intuition is that if π does have arbitrage $\pi(\mathbf{Q}) > \sum_{i=1}^{m} \pi(\mathbf{Q}_i)$, then a buyer would never pay the full price $\pi(\mathbf{Q})$; instead he would purchase $\mathbf{Q}_1, \ldots, \mathbf{Q}_m$ and use these answers to compute $\mathbf{Q}$.

Example 3.8. Consider a query $(\mathbf{q}, v)$ offered for price $\pi(\mathbf{q}, v)$. A buyer who wishes to improve the accuracy of the query may ask the same query n times, $(\mathbf{q}, v)$, $(\mathbf{q}, v), \ldots, (\mathbf{q}, v)$, at a total cost of $n \cdot \pi(\mathbf{q}, v)$. The buyer then computes the average of the query answers to get an estimated answer with a much lower variance, namely v/n. The pricing function must ensure that the total payment collected from the buyer covers the cost of this lower variance, in other words, $n \cdot \pi(\mathbf{q}, v) \geq \pi(\mathbf{q}, v/n)$. If π is arbitrage free, then it is easy to check that this condition holds. Indeed, $\{(\mathbf{q}, v), \ldots, (\mathbf{q}, v)\} \to (n\mathbf{q}, nv) \to (\mathbf{q}, v/n)$, and arbitrage freeness implies $\pi(\mathbf{q}, v/n) \leq \pi(\mathbf{q}, v) + \cdots + \pi(\mathbf{q}, v) = n \cdot \pi(\mathbf{q}, v)$.

We prove that any arbitrage free pricing function satisfies the following simple properties.

PROPOSITION 3.9. *Let π be an arbitrage-free pricing function. Then:*

(1) the zero query is free: $\pi(\mathbf{0}, v) = 0$;
(2) higher variance is cheaper: $v \leq v'$ implies $\pi(\mathbf{q}, v) \geq \pi(\mathbf{q}, v')$;
(3) the zero-variance query is the most expensive[3]: $\pi(\mathbf{q}, 0) \geq \pi(\mathbf{q}, v)$ for all $v \geq 0$; and
(4) infinite noise is free: if π is continuous at $\mathbf{q} = 0$, then $\pi(\mathbf{q}, \infty) = 0$.

PROOF. For (1), we have $\emptyset \to (\mathbf{0}, 0)$ by the first rule of Definition 3.4 (taking $k = 0$, i.e., $\mathbf{S} = \emptyset$) and $(\mathbf{0}, 0) \to (\mathbf{0}, v)$ by the third rule; hence $\pi(\mathbf{0}, v) = 0$. And (2) follows from $(\mathbf{q}, v) \to (\mathbf{q}, v')$ when $v \leq v'$ and (3) follows immediately from (2), since all variances are $v \geq 0$. For (4), we use the second rule to derive $(1/c \cdot \mathbf{q}, v) \to (\mathbf{q}, c^2 \cdot v)$, hence

$$\pi(\mathbf{q}, \infty) = \lim_{c \to \infty} \pi(\mathbf{q}, c^2 \cdot v) \leq \lim_{c \to \infty} \pi(1/c \cdot \mathbf{q}, v) = \pi(\mathbf{0}, v) = 0. \quad \square$$

Arbitrage-free pricing functions have been studied before [Koutris et al. 2012; Li and Miklau 2012], but only in the context of deterministic (i.e., unperturbed) query answers. Our definition extends those in Koutris et al. [2012] and Li and Miklau [2012] to queries with perturbed answers.

3.4. Arbitrage-Free Pricing Functions: Synthesis

The zero-cost pricing function $\pi(\mathbf{Q}) = 0$ for all $\mathbf{Q}$ is arbitrage free; this is a trivial price under which every query is free. It is not clear whether a nontrivial arbitrage-free pricing function exists at all. For example, we note that the constant-price pricing function is not arbitrage free. More precisely, fix any constant $c > 0$; the market maker charges the fixed price $\pi(\mathbf{Q}) = c$ for every query $\mathbf{Q}$. This function has arbitrage. Indeed, here the market maker also intends to charge the price $c > 0$ for the zero query $\mathbf{q}(\mathbf{x}) = 0$. But the zero query should be free, because any buyer can derive it himself (using $m = 0$ in Definition 3.7); this is also stated in Proposition 3.9 (1). Thus, it is not clear at all how to construct any nontrivial arbitrage-free pricing function. In this section we prove that a nontrivial arbitrage-free pricing functions exists, and give some sufficient rules for synthesizing such functions; in Section 3.5 we extend these rules

[3]It is possible that $\pi(\mathbf{q}, 0) = \infty$.

to allow the pricing function to set a finite price for the raw, unperturbed data; and in Section 3.6 we show that general construction of arbitrage-free pricing functions remains a difficult problem.

We start by analyzing how an arbitrage-free pricing function $\pi(\mathbf{q}, v)$ depends on the variance v. We already know that it is monotonically decreasing in v (Proposition 3.9 (2)) and that, assuming it is continuous and nontrivial, cannot be independent of v (Proposition 3.9 (4)). The next proposition shows that it cannot decrease faster than $1/v$.

PROPOSITION 3.10. *For any arbitrage-free pricing function π and any linear query $\mathbf{q}$, $\pi(\mathbf{q}, v) = \Omega(1/v)$.*

PROOF. Suppose the contrary that there exists a linear query $\mathbf{q}$ and a sequence $\{v_i\}_{i=1}^{\infty}$ such that $\lim_{i\to\infty} v_i = +\infty$ and $\lim_{i\to\infty} v_i\pi(\mathbf{q}, v_i) = 0$. Select i_0 such that $v_{i_0} > 1$ and $v_{i_0}\pi(\mathbf{q}, v_{i_0}) < \pi(\mathbf{q}, 1)/2$. Then, we can answer $\pi(\mathbf{q}, 1)$ by asking the query $\pi(\mathbf{q}, v_{i_0})$ at most $\lceil v_{i_0} \rceil$ times and computing the average. For these $\lceil v_{i_0} \rceil$ queries we pay

$$\lceil v_{i_0} \rceil \pi(\mathbf{q}, v_{i_0}) \le (v_{i_0} + 1)\pi(\mathbf{q}, v_{i_0}) < 2v_{i_0}\pi(\mathbf{q}, v_{i_0}) < \pi(\mathbf{q}, 1),$$

which implies that we have arbitrage, a contradiction. □

The first arbitrage-free pricing function that we synthesize is $\pi(\mathbf{q}, v) = f^2(\mathbf{q})/v$, for some positive function f that depends only on $\mathbf{q}$. We prove π is arbitrage free iff f is a seminorm. This gives us an entire class of arbitrage-free pricing functions, as well as a complete characterization of those functions whose dependency on v is $\Theta(1/v)$. Recall that a *semi-norm* is a function $f : \mathbb{R}^n \to \mathbb{R}$ that satisfies the following properties[4]:

—for any $c \in \mathbb{R}$ and any $\mathbf{q} \in \mathbb{R}^n$, $f(c\mathbf{q}) = |c|\, f(\mathbf{q})$; and
—for any $\mathbf{q}_1, \mathbf{q}_2 \in \mathbb{R}^n$, $f(\mathbf{q}_1 + \mathbf{q}_2) \le f(\mathbf{q}_1) + f(\mathbf{q}_2)$.

We prove the following.

THEOREM 3.11. *Let $\pi(\mathbf{q}, v)$ be a pricing function such that $\pi(\mathbf{q}, v) = f^2(\mathbf{q})/v$ for some function f.[5] Then $\pi(\mathbf{q}, v)$ is arbitrage free iff $f(\mathbf{q})$ is a semi-norm.*

PROOF. ($\Leftarrow$) : Suppose $\pi(\mathbf{q}, v) = f^2(\mathbf{q})/v$ and $f(\mathbf{q})$ is a semi-norm. According to Proposition 3.5, $\{(\mathbf{q}_1, v_1), \ldots, (\mathbf{q}_m, v_m)\} \to (\mathbf{q}, v)$ if and only if there exists $c_1, \ldots, c_m$ such that $c_1\mathbf{q}_1 + \cdots + c_m\mathbf{q}_m = \mathbf{q}$ and $c_1^2 v_1 + \cdots + c_m^2 v_m \le v$. Then,

$$\begin{aligned}
\sum_{i=1}^{m} \pi(\mathbf{q}_i, v_i) &= \sum_{i=1}^{m} \frac{f^2(\mathbf{q}_i)}{v_i} = \frac{\left(\sum_{i=1}^{m} \frac{f^2(\mathbf{q}_i)}{v_i}\right)\left(\sum_{i=1}^{m} c_i^2 v_i\right)}{\sum_{i=1}^{m} c_i^2 v_i} \\
&\ge \frac{\left(\sum_{i=1}^{m} |c_i|\, f(\mathbf{q}_i)\right)^2}{\sum_{i=1}^{m} c_i^2 v_i} = \frac{\left(\sum_{i=1}^{m} f(c_i\mathbf{q}_i)\right)^2}{\sum_{i=1}^{m} c_i^2 v_i} \\
&\ge \frac{f(\mathbf{q})^2}{v} = \pi(\mathbf{q}, v),
\end{aligned}$$

where the first inequality follows from the Cauchy-Schwarz inequality and the second from the subadditivity of the semi-norm.

[4]It follows that $f(\mathbf{0}) = 0$ (by taking $c = 0$ in the first property); if the converse holds, that is, $f(\mathbf{q}) = 0 \Rightarrow \mathbf{q} = \mathbf{0}$, then f is called a *norm*. Also, recall that any semi-norm satisfies $f(\mathbf{q}) \ge 0$, by the triangle inequality.
[5]In other words, $f(\mathbf{q}) = \sqrt{\pi(\mathbf{q}, v)v}$ is independent of v.

($\Rightarrow$) : Assuming π is arbitrage free, we prove that f is a semi-norm. For $c \neq 0$, by the second rule of Definition 3.4, we have both

$$(\mathbf{q}, v) \rightarrow (c\mathbf{q}, c^2 v),$$
$$(c\mathbf{q}, c^2 v) \rightarrow \left(\frac{1}{c} \times c\mathbf{q}, \left(\frac{1}{c}\right)^2 \times c^2 v\right) \rightarrow (\mathbf{q}, v),$$

therefore both $\pi(\mathbf{q}, v) \leq \pi(c\mathbf{q}, c^2 v)$ and $\pi(\mathbf{q}, v) \geq \pi(c\mathbf{q}, c^2 v)$ hold, thus $\pi(\mathbf{q}, v) = \pi(c\mathbf{q}, c^2 v)$. This implies that, if $c \neq 0$,

$$f(c\mathbf{q}) = \sqrt{\pi(c\mathbf{q}, c^2 v) c^2 v} = |c| \sqrt{\pi(\mathbf{q}, v) v} = |c| f(\mathbf{q}).$$

If $c = 0$, we also have $f(c\mathbf{q}) = \sqrt{\pi(c\mathbf{q}, c^2 v) c^2 v} = 0 = |c| f(\mathbf{q})$.

Next we prove that $f(\mathbf{q}_1 + \mathbf{q}_2) \leq f(\mathbf{q}_1) + f(\mathbf{q}_2)$. Set the variances $v_1 = f(\mathbf{q}_1)$ and $v_2 = f(\mathbf{q}_2)$; then we have $f(\mathbf{q}_1) = \pi(\mathbf{q}_1, v_1)$ and $f(\mathbf{q}_2) = \pi(\mathbf{q}_2, v_2)$. By the first rule in Definition 3.4 we have $\{(\mathbf{q}_1, v_1), (\mathbf{q}_2, v_2)\} \rightarrow (\mathbf{q}_1 + \mathbf{q}_2, v_1 + v_2)$, and therefore

$$\begin{aligned} \frac{f^2(\mathbf{q}_1 + \mathbf{q}_2)}{f(\mathbf{q}_1) + f(\mathbf{q}_2)} &= \pi(\mathbf{q}_1 + \mathbf{q}_2, v_1 + v_2), \\ &\leq \pi(\mathbf{q}_1, v_1) + \pi(\mathbf{q}_2, v_2) = f(\mathbf{q}_1) + f(\mathbf{q}_2), \end{aligned}$$

which proves the claim. □

As an immediate application of the theorem, let us instantiate f to be one of the norms L_2, L_∞, L_p, or a weighted L_2 norm. This implies that the following four functions are arbitrage free:

$$\pi(\mathbf{q}, v) = ||\mathbf{q}||_2^2 / v = \sum_i q_i^2 / v, \tag{2}$$

$$\pi(\mathbf{q}, v) = ||\mathbf{q}||_\infty^2 / v = \max_i q_i^2 / v, \tag{3}$$

$$\pi(\mathbf{q}, v) = ||\mathbf{q}||_p^2 / v = \left(\sum_i q_i^p\right)^{2/p} / v \qquad p \geq 1, \tag{4}$$

$$\pi(\mathbf{q}, v) = \left(\sum_i w_i \cdot q_i^2\right) / v \qquad w_1, \ldots, w_n \geq 0. \tag{5}$$

Next, we give a general method for synthesizing new arbitrage-free pricing functions from existing ones. Recall that a function $f : (\bar{\mathbb{R}}^+)^k \rightarrow \bar{\mathbb{R}}^+$ is called *subadditive* if, for any two vectors $\mathbf{x}, \mathbf{y} \in (\bar{\mathbb{R}}^+)^k$, $f(\mathbf{x}+\mathbf{y}) \leq f(\mathbf{x}) + f(\mathbf{y})$; the function is called *nondecreasing* if $\mathbf{x} \leq \mathbf{y}$ implies $f(\mathbf{x}) \leq f(\mathbf{y})$.

PROPOSITION 3.12. *Let $f : (\bar{\mathbb{R}}^+)^k \rightarrow \bar{\mathbb{R}}^+$ be a subadditive, nondecreasing function. For any arbitrage-free pricing functions $\pi_1, \ldots, \pi_k$, the function $\pi(\mathbf{Q}) = f(\pi_1(\mathbf{Q}), \ldots, \pi_k(\mathbf{Q}))$ is also arbitrage free.*

PROOF. For any query $\mathbf{Q}$, let $\bar{\pi}(\mathbf{Q}) = (\pi_1(\mathbf{Q}), \ldots, \pi_k(\mathbf{Q}))$. Assume $\{(\mathbf{q}_1, v_1), \ldots, (\mathbf{q}_m, v_m)\} \to (\mathbf{q}, v)$. We have

$$\begin{aligned}
\bar{\pi}(\mathbf{Q}) &\le \sum_i \bar{\pi}(\mathbf{Q}_i) && \text{because each } \pi_j \text{ is arbitrage free,}\\
f(\bar{\pi}(\mathbf{Q})) &\le f\left(\sum_i \bar{\pi}(\mathbf{Q}_i)\right) && \text{because } f \text{ is nondecreasing,}\\
&\le \sum_i f(\bar{\pi}(\mathbf{Q}_i)) && \text{because } f \text{ is subadditive.} \quad \square
\end{aligned}$$

Proposition 3.12 allows us to synthesize a new arbitrage-free pricing function from existing ones. As an application we obtain the next corollary.

COROLLARY 3.13. *If $\pi_1, \ldots, \pi_k$ are arbitrage-free pricing functions, then so are the following functions:*

—linear combination: $c_1\pi_1 + \cdots + c_k\pi_k$, for $c_1, \ldots, c_k \ge 0$;
—maximum: $max(\pi_1, \ldots, \pi_k)$;
—cut-off: $min(\pi_1, c)$, where $c \ge 0$;
—power: π_1^c where $0 < c \le 1$;
—logarithmic: $log(\pi_1 + 1)$; and
—geometric mean: $\sqrt{\pi_1 \cdot \pi_2}$.

PROOF. It is clear that all the preceding functions are monotonically increasing. One can check directly that maximum and cut-off functions are subadditive. Subadditivity for the rest follows from the following.

LEMMA 3.14. *Let $f : (\bar{\mathbb{R}}^+)^k \to \bar{\mathbb{R}}^+$ be a nondecreasing, continuous, and twice-differentiable function such that $f(\mathbf{0}) = 0$. Then, if $\partial^2 f/\partial x_i \partial x_j \le 0$ for all $i, j = 1, \ldots, k$, then f is subadditive.*

PROOF. Denote $f_i = \partial f/\partial x_i$ and $f_{ij} = \partial^2 f/\partial x_i \partial x_j$. We apply twice the first-order Taylor approximation $f(\mathbf{x}) - f(\mathbf{0}) = \sum_i (\partial f/\partial x_i)(\xi) \cdot x_i$, once to $g(\mathbf{y}) = f(\mathbf{x} + \mathbf{y}) - f(\mathbf{y})$, and the second time to $h(\mathbf{x}) = \sum_j (f_j(\mathbf{x} + \xi) - f_j(\xi)) \cdot y_j$:

$$\begin{aligned}
f(\mathbf{x}) + f(\mathbf{y}) - f(\mathbf{x} + \mathbf{y}) &= [f(\mathbf{x}) - f(\mathbf{0})] + [f(\mathbf{x} + \mathbf{y}) - f(\mathbf{y})]\\
&= g(\mathbf{0}) - g(\mathbf{y}) = -\sum_j g_j(\xi) \cdot y_j\\
&= -\sum_j (f_j(\mathbf{x} + \xi) - f_j(\xi)) \cdot y_j = -\sum_{ij} f_{ij}(\eta + \xi) \cdot x_i \cdot y_j \ge 0. \quad \square
\end{aligned}$$

Example 3.15. For a simple illustration we will prove that the pricing function $\pi(\mathbf{q}, v) = \max_i |q_i|/\sqrt{v}$ is arbitrage free. Start from $\pi_1(\mathbf{q}, v) = \max_i q_i^2/v$, which is arbitrage free by Eq. (3), then notice that $\pi = (\pi_1)^{1/2}$, hence π is arbitrage free by Corollary 3.13.

3.5. Raw Private Data at a Finite Price

While under differential privacy perturbation is always necessary, in data markets the data being sold is usually unperturbed. Perturbation is only a tool to reduce the price for the buyer. Therefore, a reasonable pricing function $\pi(\mathbf{q}, v)$ needs to give a finite price for a zero variance; all functions that we described in Section 3.4, with the exception of the cut-off function, charge an infinite price for the raw data. In this section we discuss pricing functions that set a finite price for the raw private data.

For this, we simply use subadditive, nondecreasing functions that are bounded and apply Proposition 3.12. In addition to the cut-off function, we can use sigmoid curves (popular in the machine learning community) and inverse trigonometric functions.

COROLLARY 3.16. *Given an arbitrage-free pricing function π, each of the following functions is also arbitrage free and bounded:* $\mathtt{atan}(\pi)$, $\mathtt{tanh}(\pi)$, $\pi/\sqrt{\pi^2+1}$.

PROOF. By Lemma 3.14 it suffices to check that all first derivatives ≥ 0 and all second derivatives ≤ 0, for all $x \geq 0$:

$$\frac{d}{dx}\mathtt{atan}(x) = \frac{1}{1+x^2} > 0;$$
$$\frac{d^2}{dx^2}\mathtt{atan}(x) = -\frac{2x}{(1+x^2)^2} \leq 0;$$
$$\frac{d}{dx}\mathtt{tanh}(x) = \frac{1}{\mathtt{cosh}^2(x)} > 0;$$
$$\frac{d^2}{dx^2}\mathtt{tanh}(x) = -\frac{2\mathtt{tanh}(x)}{\mathtt{cosh}^2(x)} \leq 0;$$
$$\frac{d}{dx}\frac{x}{\sqrt{1+x^2}} = (1+x^2)^{-\frac{3}{2}} > 0;$$
$$\frac{d^2}{dx^2}\frac{x}{\sqrt{1+x^2}} = -3x(1+x^2)^{-\frac{5}{2}} \leq 0. \quad \square$$

Example 3.17. Suppose we want to charge a price p for the true, unperturbed result of a query $\mathbf{q}$. Assume $||\mathbf{q}||_2^2 = n$, and let $\pi_1(\mathbf{q}, v) = ||\mathbf{q}||_2^2/v = n/v$ be the pricing function in Eq. (2). It follows that the function[6]

$$\pi(\mathbf{q}, v) = \frac{2p}{\Pi} \cdot \mathtt{atan}(c \cdot \pi_1(\mathbf{q}, v)) = \frac{2p}{\Pi} \cdot \mathtt{atan}\left(c\frac{n}{v}\right)$$

is arbitrage free. Here $c > 0$ is a parameter. For example, suppose the buyer cannot afford the unperturbed query ($v = 0$), and settles instead for a variance $v = \Theta(n)$ (it corresponds to a standard deviation $\sqrt{n}$, which is sufficient for some applications); for concreteness, assume $v = 5n$. Then $\pi(\mathbf{q}, v) = \frac{2p}{\Pi} \cdot \mathtt{atan}(c/5)$. To make this price affordable, we choose $c \ll 1$, in which case the price becomes $\pi \approx 2 \cdot c \cdot p/(5 \cdot \Pi) = 0.13 \cdot c \cdot p$. In Example 2.3 the price of the unperturbed query was $p = \$10000$, and we wanted to charge \$1 for the variance $v = 5n = 5000$. For this we can use the pricing function π given before, with $c = 1/(0.13 \cdot p) = 7.85 \cdot 10^{-4}$. We can now answer the question in Example 2.3: the cost of the query with variance $v = 50$ is $\pi(\mathbf{q}, v) = \frac{2p}{\Pi} \cdot \mathtt{atan}(100 \cdot c/5) = \$99.94$.

3.6. Deriving Pricing Functions from Price Points

We discuss here a more flexible framework for defining an arbitrage-free pricing function: the market maker defines the price for a finite set of queries called *views*, then the system automatically extrapolates this to a pricing function on all queries.

Definition 3.18. A *price point* is a pair $(\mathbf{V}, p)$, where $\mathbf{V} = (\mathbf{q}, v)$ is a query (called a view) and $p \in \mathbb{R}^+$ is a fixed price.

[6]We use Π for the constant 3.1415... to avoid confusion with the pricing function π.

Koutris et al. [2012] introduce a pricing framework where the market maker defines a set of price points $\mathcal{V} = ((\mathbf{V}_1, p_1), \ldots, (\mathbf{V}_m, p_m))$, and the system synthesizes an arbitrage-free pricing function π that agrees with all price points. This is an ideal way to set prices; Koutris et al. [2012] give necessary and sufficient conditions for such a pricing function to exist, but only for relational queries, without perturbation. In our framework, the queries are linear and include random noise; we show here that the techniques in Koutris et al. [2012] can be adapted to our setting, and prove that computing a price function given by a set of price points is NP-complete.

We start with the following definition, adapted from Koutris et al. [2012].

Definition 3.19. Given a set of price points $\mathcal{V} = ((\mathbf{V}_1, p_1), \ldots, (\mathbf{V}_m, p_m))$, we say that a pricing function π is *valid* with respect to $\mathcal{V}$ if (a) π is arbitrage free, and (b) $\forall(\mathbf{V}_i, p_i) \in \mathcal{V}$, $\pi(\mathbf{V}_i) = p_i$.

In general, there may not exist a valid pricing function. For example, assume that $\mathbf{V}_1, \ldots, \mathbf{V}_{m-1}$ determine $\mathbf{V}_m$, and assume that $p_m > p_1 + \cdots + p_{m-1}$. Then, there is no valid pricing function because we would have a contradiction; condition (b) implies $\pi(\mathbf{V}_m) = p_m$, while (a) implies $p_m = \pi(\mathbf{V}_m) \leq \sum_{i=1,m-1} \pi(\mathbf{V}_i) = \sum_{i=1,m-1} p_i$.

If a valid pricing function exists, then we say $\mathcal{V}$ is *consistent*.

Note that, if $\mathcal{V}$ is consistent, then it cannot contain the same query twice with different prices: if $(\mathbf{V}, p_i), (\mathbf{V}, p_j) \in \mathcal{V}$ and $p_i \neq p_j$ then there is arbitrage. We will assume without loss of generality that the queries $\mathbf{V}_1, \ldots, \mathbf{V}_m$ are distinct.

The key to designing a valid pricing function is to compute, for any given query $\mathbf{Q}$, the cheapest plan to answer $\mathbf{Q}$ from the views in $\mathcal{V}$. A *procurement plan* is a plan for answering $\mathbf{Q}$ that specifies how many times to purchase from the market maker each query $\mathbf{V}_i$.

Definition 3.20 (*Procurement Plan*). Consider an ordered set of price points $\mathcal{V} = \{(\mathbf{V}_1, p_1), (\mathbf{V}_2, p_2), \ldots, (\mathbf{V}_m, p_m)\}$. A *procurement plan* is an ordered multiset of nonnegative integers $\mathbf{B} = \{b_1, b_2, \ldots, b_m\}$ such that the multiset $\mathcal{V}^{\mathbf{B}} = \{\mathbf{V}_1^{b_1}, \ldots, \mathbf{V}_m^{b_m}\}$ (where each query $\mathbf{V}_i$ occurs b_i times) determines $\mathbf{Q}$: $\mathcal{V}^{\mathbf{B}} \to \mathbf{Q}$. The *cost* of the procurement plan is $\text{cost}(\mathbf{B}) = \sum_i b_i p_i$.

$\mathcal{V}$ and $\mathbf{B}$ need to be ordered so as to establish a one-to-one correspondence between $(\mathbf{V}_i, p_i)$ and $\mathbf{b}_i$. For a trivial example, let $\mathcal{V} = \{((\mathbf{q}, 100), \$5)\}$ consist of a single price point that charges \$5 for some query with variance 100. The buyer wants to purchase $\mathbf{Q} = (\mathbf{q}, 25)$; a procurement plan is $b_1 = 4$, in other words, we must purchase the query four times and compute the average in order to reduce the variance to 25. Obviously, this is the cheapest procurement plan for $\mathbf{Q}$.

Unlike in the definition of answerability $\mathbf{S} \to \mathbf{Q}$ (Definition 3.3) where we had one output for each query in $\mathbf{S}$ and could only use these outputs to answer $\mathbf{Q}$, in a procurement plan we can buy a view $\mathbf{V}_i$ as many times as we want, for example, in order to reduce its variance. Of course, we have to pay for it, and this is captured by cost. A query $\mathbf{Q} = (\mathbf{q}, v)$ has a procurement plan iff $\mathbf{q}$ is a linear combination, and this can be checked by solving a system of linear equations; in other words, the variance doesn't matter for the *existence* of a procurement plan, since we can always compute the query repeatedly and take the average, thus reducing the variance.[7] Of course, we want to find the procurement plan of lowest cost, which justifies the following.

[7]The only exception is when the query has variance zero, $v = 0$. Since averaging can only reduce the variance but never make it zero, in such a case we can only use the views in $\mathcal{V}$ whose variance is zero. In other words, a procurement plan exists iff $\mathbf{q}$ is a linear combination of those views in $\mathcal{V}$ that have a zero variance.

Definition 3.21. Given a set of price points $\mathcal{V}$, the *arbitrage pricing function* is

$$\pi_{\mathcal{V}}(\mathbf{Q}) = \min\{\text{cost}(\mathbf{B}) \mid \mathcal{V}^{\mathbf{B}} \to \mathbf{Q}.\}.$$

The arbitrage function is very intuitive. It takes as input a query $\mathbf{Q}$ and finds the cheapest procurement plan $\mathbf{B}$ for $\mathbf{Q}$; in effect, it performs "arbitrage" in order to answer $\mathbf{Q}$, then returns the cost of that plan. Note that, by definition, if $\mathbf{Q}$ is not answerable from $\mathcal{V}$, then the arbitrage price is $\pi_{\mathcal{V}}(\mathbf{Q}) = \infty$.

The following theorem extends a similar property in Koutris et al. [2012, Theorem 2.15].

THEOREM 3.22. *Let $\mathcal{V}$ be a set of price points and $\pi_{\mathcal{V}}$ its arbitrage function. Then: (a) $\pi_{\mathcal{V}}$ is arbitrage free; and (b) if π is any valid pricing function for $\mathcal{V}$, then $\pi(\mathbf{Q}) \le \pi_{\mathcal{V}}(\mathbf{Q})$ for all queries $\mathbf{Q}$; and (c) $\mathcal{V}$ is consistent iff for all $(\mathbf{V}_i, p_i) \in \mathcal{V}$, $p_i \le \pi_{\mathcal{V}}(V_i)$.*

PROOF. (a) Suppose $\{\mathbf{Q}_1, \mathbf{Q}_2, \mathbf{Q}_3, \ldots, \mathbf{Q}_k\} \to \mathbf{Q}$; we prove that $\sum_{i=1,k} \pi_{\mathcal{V}}(\mathbf{Q}_i) \ge \pi_{\mathcal{V}}(\mathbf{Q})$. By definition, $\pi_{\mathcal{V}}(\mathbf{Q}) = \min_{\mathbf{B}:\mathbf{V}^{\mathbf{B}} \to \mathbf{Q}} \sum_{j=1,m} b_j p_j$, and $\pi_{\mathcal{V}}(\mathbf{Q}_i) = \min_{\mathbf{B}:\mathbf{V}^{\mathbf{B}} \to \mathbf{Q}_i} \sum_{j=1,m} b_j p_j$, $i \in \{1, 2, \ldots, k\}$.

Let $\mathbf{B}_i = \text{argmin}_{\mathbf{B}:\mathbf{V}^{\mathbf{B}} \to \mathbf{Q}_i} \sum_{j=1,m} b_j p_j$. Then we know that $\mathbf{V}^{\mathbf{B}_i} \to \mathbf{Q}_i$. Since $\{\mathbf{Q}_1, \mathbf{Q}_2, \mathbf{Q}_3, \ldots, \mathbf{Q}_k\} \to \mathbf{Q}$, by the transitivity rule in Definition 3.4, $\cup_i \mathbf{V}^{\mathbf{B}_i} \to \mathbf{Q}$; here $\cup_i \mathbf{V}^{\mathbf{B}_i}$ represents multiset union. Let $\sum_i \mathbf{B}_i$ denote the procurement plan that adds the multiplicities of all $\mathbf{B}_i$'s componentwise; then $\cup_i \mathbf{V}^{\mathbf{B}_i} = \mathbf{V}^{\sum_i \mathbf{B}_i}$ and therefore $\sum_i \mathbf{B}_i$ is a procurement plan for $\mathbf{Q}$ with $\text{cost}(\sum_i \mathbf{B}_i) = \sum_i \text{cost}(\mathbf{B}_i) = \sum_i \pi_{\mathcal{V}}(\mathbf{Q}_i)$. By the definition of the arbitrage function, we have $\pi_{\mathcal{V}}(\mathbf{Q}) \le \text{cost}(\sum_i \mathbf{B}_i)$, which proves claim (a).

(b) For any query $\mathbf{Q}$, let $\mathbf{B}$ be the lowest-cost procurement plan, thus $\pi_{\mathcal{V}}(\mathbf{Q}) = \text{cost}(\mathbf{B})$. Since $\mathcal{V}^{\mathbf{B}} \to \mathbf{Q}$, for any valid pricing function π we must have $\pi(\mathbf{Q}) \le b_1\pi(\mathbf{V}_1) + \cdots + b_m\pi(\mathbf{V}_m)$ (since π is arbitrage free), therefore $\pi(\mathbf{Q}) \le b_1 p_1 + \cdots b_m p_m = \text{cost}(\mathbf{B})$ (since π is valid), and claim (b) follows from $\text{cost}(\mathbf{B}) = \pi_{\mathcal{V}}(\mathbf{Q})$.

(c) We note that, for all $(\mathbf{V}_i, p_i) \in \mathcal{V}$, $\pi_{\mathcal{V}}(\mathbf{V}_i) \le p_i$, because there exists a trivial procurement plan for $\mathbf{V}_i$ that purchases $\mathbf{V}_i$ exactly once, and its cost is p_i. Therefore, if $p_i \le \pi_{\mathcal{V}}(V_i)$, then $\pi_{\mathcal{V}}$ is valid for $\mathcal{V}$, proving $\mathcal{V}$ is consistent. Conversely, if $\mathcal{V}$ is consistent then there exists a valid pricing function π, and by (b) we have $p_i = \pi(\mathbf{V}_i) \le \pi_{\mathcal{V}}(\mathbf{Q})$. □

This suggests a simple method for a market maker to set almost arbitrary prices: the market maker defines a set of price points, $\mathcal{V} = \{(\mathbf{V}_i, p_i) \mid i = 1, m\}$, the system checks consistency by checking $p_i \le \pi_{\mathcal{V}}(\mathbf{V}_i)$ for all i, then the market maker sells every query $\mathbf{Q}$ at price $\pi_{\mathcal{V}}(\mathbf{Q})$. Unfortunately, we show this procedure has high complexity, both during the consistency check step and during the price computation step.

THEOREM 3.23. *The following problem: "given a set of price points $\mathcal{V}$, a query $\mathbf{Q}$, and budget $p > 0$, decide whether $\mathbf{Q}$ is answerable from $\mathcal{V}$ within a budget p (in other words, check $\pi_{\mathcal{V}}(\mathbf{Q}) \le p$)", is NP-complete.*

Before we prove the theorem, we make two comments. First, it is interesting to contrast the theorem to Proposition 3.6. There, we have shown that checking determinacy is in PTIME(m, n). The difference is that, in that case we had a fixed set of answers to the queries in $\mathbf{S}$, and the problem reduced to a quadratic program. Now, we also need to decide how many times to purchase each view in $\mathcal{V}$, and the problem is a combinatorial optimization. Second, we contrast this to the complexity of computing the arbitrage function for relational queries in Koutris et al. [2012]. While the complexity was also high in that setting, the problem became tractable by imposing natural restrictions on the query language, whereas in our case the queries are already very simple (linear queries), and there are no obvious restrictions to make the problem tractable.

In the rest of this section we give the proof of the theorem. We use a reduction to the following NP-complete problem.

Definition 3.24 (*Exact Cover by* 3-*Sets*).
Instance: Set $\mathbf{X}$ with $|\mathbf{X}| = 3d$ and a collection $\mathbf{C}$ of three-element subsets of X.
Problem: Does $\mathbf{C}$ contain an exact cover for $\mathbf{X}$, that is, a subcollection $\mathbf{C}' \subseteq \mathbf{C}$ such that every element of $\mathbf{X}$ occurs in exactly one member of $\mathbf{C}'$?

The exact cover by 3-sets problem (X3C) is shown to be NP-complete [Garey and Johnson 1979].

PROOF. First we show that the problem is in NP. Given a procurement plan $\mathbf{B}$, one can check $\mathcal{V}^{\mathbf{B}} \to Q$ in PTIME(m, n) by Proposition 3.6. The claim follows by observing that each number b_i in the procurement plan is bounded by $\lfloor p/p_i \rfloor$ when the variance $v_i > 0$, and by 1 when $v_i = 0$ (since we only need to purchase once any zero-variance query).

We prove NP-hardness by reduction from the exact cover by 3-sets problem.

Consider an instance of the exact cover by three-sets problem, $\mathbf{X} = \{x_1, x_2, \ldots, x_n\}$, with $n = 3d$, and a collection $\mathbf{C} = \{\mathbf{C}_1, \mathbf{C}_2, \ldots, \mathbf{C}_m\}$ of three-element subsets of $\mathbf{X}$, we transform it into an instance of the budget constraint answerability problem as follows.

To each subset $\mathbf{C}_j$, $j = 1, \ldots, m$, we associate a query $\mathbf{q}_j$, which is an n-dimensional vector whose i'th dimension is 1 if x_i belongs to $\mathbf{C}_j$, and 0 otherwise. Define the price points $\mathcal{V} = \{((q_j, 0), 1) \mid j = 1, \ldots, m\}$; they contain each query q_j, with variance 0, and price 1. Let $\mathbf{q} = (1, 1, \ldots, 1)$ be an n-dimensional vector of all 1's, $\mathbf{Q} = (\mathbf{q}, 0)$, and $p = d + 0.5$.

We will show that $\mathbf{X}$ has an exact cover if and only if $\mathbf{Q}$ is answerable from $\mathcal{V}$ within a budget p.

(Budget constraint answerability to Exact Cover by 3-Sets): If $\mathbf{Q}$ is answerable from $\mathcal{V}$ within a budget p, then there is a procurement plan $\mathbf{B} = (b_1, b_2, \ldots, b_m)$ containing exactly k nonzero entries $b_{i_1}, b_{i_2}, \ldots, b_{i_k}$, and proper assignments to $c_{i_1}, c_{i_2}, \ldots, c_{i_k}$ such that $c_{i_1}\mathbf{q}_{i_1} + c_{i_2}\mathbf{q}_{i_2} + \cdots + c_{i_k}\mathbf{q}_{i_k} = \mathbf{q}$, and $p_{i_1}b_{i_1} + p_{i_2}b_{i_2} + \cdots + p_{i_k}b_{i_k} \leq p = d + 0.5$. Since the quantity on the left is an integer, it must be $\leq d$. On the other hand, due to the fact that each $\mathbf{C}_j$, $j \in \{1, \ldots, m\}$ only has three elements and $c_{i_1}\mathbf{q}_{i_1} + c_{i_2}\mathbf{q}_{i_2} + \cdots + c_{i_k}\mathbf{q}_{i_k} = \mathbf{q}$, we have $k \geq d$. Therefore $k = d$, and $c_{i_1} = c_{i_2} = \cdots = c_{i_k} = 1$. In other words, $\mathbf{q}_{i_1} + \mathbf{q}_{i_2} + \cdots + \mathbf{q}_{i_k} = \mathbf{q}$, meaning $\mathbf{C}_{i_1}, \mathbf{C}_{i_2}, \ldots, \mathbf{C}_{i_k}$ can exactly cover $\mathbf{X}$.

(Exact Cover by 3-Sets to Budget constraint answerability): If there is an exact cover of 3-sets for $\mathbf{X}$, say $\mathbf{C}_{i_1}, \mathbf{C}_{i_2}, \ldots, \mathbf{C}_{i_d}$, then $\mathbf{q}_{i_1} + \mathbf{q}_{i_2} + \cdots + \mathbf{q}_{i_d} = \mathbf{q}$. The procurement plan defined by $b_{i_1} = b_{i_2} = \cdots = b_{i_d} = 1$ and $b_i = 0$ for all $i \notin \{i_1, \ldots, i_d\}$ has cost $d \leq p$, proving the claim. □

3.7. Discussion

We discuss here two important restrictions of our framework that we hope may be relaxed in future work on the topic. First, we have assumed users are interested only in *unbiased* estimators. The market maker is required to answer queries using an unbiased estimator (Definition 3.2) and arbitrage freeness only prevents an adversary from deriving an unbiased estimator of a query using cheaper queries (Definition 3.3). Yet the literature contains several examples of biased estimators that perform better than unbiased ones, for example, SQ learning algorithms [Blum 2003] or the Hardt-Rothblum estimators for linear queries [Hardt and Rothblum 2010].

A more general approach would be to relax Definition 3.2 to allow queries to be answered using biased or unbiased estimators. This would provide the market maker with a stronger guarantee against arbitrage. However, it would likely make reasoning about determinacy and arbitrage-free pricing more difficult, and further restrict the

set of arbitrage-free pricing functions available to the market maker. In other words, a more powerful notion of arbitrage would lead to more restrictive pricing functions, potentially limiting the ability of the market maker to set prices. Exploring these trade-offs for biased answerability is an important direction for future work and may be a requirement for practical instances of our framework.

Second, we have considered a restricted notion of answerability, limiting our attention to adversaries that only consider queries computed by linear functions of other queries. A real adversary may be able to answer a query using a nonlinear function of precomputed queries, or may use external information to compute the answer. Again, this assumption limits the class of adversaries against which the arbitrage-free property is guaranteed to hold. For example, there exists an unbiased nonlinear estimator whose variance is smaller than linear estimators [Knautz 1999] when the noise distribution is not Gaussian. We made this assumption primarily for technical reasons (because we can currently characterize only linear answerability).

As previously, relaxing this assumption would further limit the available pricing functions that, as we have shown, are already difficult to synthesize. At the same time, we have granted considerable power to the adversary reasoning about linear answerability since deciding answerability is exponential in the number of dimensions of the data (see the proof of Proposition 3.6).

We believe our initial assumptions are reasonable restrictions that allow us to make initial progress on the pricing of private data in our proposed framework. However, ongoing work should address these assumptions and the trade-offs between the benefits to the market maker of considering more powerful adversaries and stronger notions of arbitrage and the practical feasibility of query pricing. Such work will need to consider the potential risks and costs of arbitrage, the complexity (for the market maker) of reasoning about arbitrage, as well as the complexity (for the adversary) of successfully discovering and exploiting arbitrage opportunities that may exist.

4. PRIVACY LOSS

In this section we describe the second component of the pricing framework in Figure 1, namely the privacy loss for each owner, denoted by ε. Recall that, for each buyer's query $\mathbf{Q} = (\mathbf{q}, v)$, the market maker defines a random function $\mathcal{K}_{\mathbf{Q}}$ such that, for any database instance $\mathbf{x}$, the random variable $\mathcal{K}_{\mathbf{Q}}(\mathbf{x})$ has expectation $\mathbf{q}(\mathbf{x})$ and variance less than or equal to v. By answering the query through this mechanism, the market maker leaks some information about each owner's data, and owners expect to be compensated appropriately. In this section we define formally the privacy loss and establish a few of its properties. In the next section we will relate the privacy loss to the micropayment that the owner expects.

Our definition of privacy loss is adapted from differential privacy, which compares the output of a mechanism with and without the contribution of one individual. Given the database vector $\mathbf{x}$, denote by $\mathbf{x}^{(j)}$ the database vector that results from adding one count to entry j and leaving all other values unchanged. That is, $\mathbf{x}^{(j)}$ represents the database with one more individual whose value matches entry j. In the differential privacy literature, pairs of such databases are referred to as *neighbors*.

Definition 4.1. Let $\mathcal{K}$ be any mechanism (meaning that, for any database instance $\mathbf{x}$, $\mathcal{K}(\mathbf{x})$ is a random variable). The *privacy loss* to each user, in notation $\varepsilon(\mathcal{K}) \in \bar{\mathbb{R}}^+$, is defined as

$$\varepsilon(\mathcal{K}) = \sup_{S,\mathbf{x},j} \left| \log \frac{\Pr\left(\mathcal{K}(\mathbf{x}) \in S\right)}{\Pr\left(\mathcal{K}(\mathbf{x}^{(j)}) \in S\right)} \right|,$$

where $\mathbf{x}$ ranges over all possible databases and S ranges over measurable sets of $\mathbb{R}$.

Revealing that one individual's tuple matches entry j leads to privacy loss, and so does revealing that an individual's value is not j. Hence one individual's privacy loss is measured by comparing the mechanism output on two data vectors that differ in *any* one entry. It follows that, for any query that is not the zero query, the privacy loss is equivalent for all users, whether or not their tuple in any particular database instance matches the query terms. We explain the connection to differential privacy in the next section. For now, we derive some simple properties of the privacy-loss function.

PROPOSITION 4.2. *Suppose $\mathcal{K}$ is a deterministic mechanism. Then $\varepsilon(\mathcal{K})$ is either* 0 *when $\mathcal{K}$ is independent of the input* $\mathbf{x}$ *or* ∞ *when $\mathcal{K}$ depends on the input* $\mathbf{x}$.

The following is well known [Dwork et al. 2006].

PROPOSITION 4.3. *Let $\mathcal{K}_1, \ldots, \mathcal{K}_m$, be mechanisms with privacy losses $\varepsilon_1, \ldots, \varepsilon_m$. Let $\mathcal{K} = f(\mathcal{K}_1, \ldots, \mathcal{K}_m)$ be a new mechanism. Then its privacy loss $\varepsilon(\mathcal{K}) \leq \varepsilon_1 + \cdots + \varepsilon_m$.*

In Section 3 we have defined the price $\pi(\mathbf{Q})$ to depend only on the variance, and not on the mechanism used to answer the query. The privacy loss (Definition 4.1), however, depends on the actual mechanism $\mathcal{K}$, and there is no general upper bound that depends only on the variance. For this purpose we will follow the techniques developed for differential privacy and will restrict the mechanism to be a Laplacian noise. In this case the privacy loss can be given as a function of the variance and of a property that depends only the query, called query sensitivity.

Definition 4.4 (*Sensitivity*). The *sensitivity* $s_{\mathbf{q}}$ of a query $\mathbf{q} = (q_1, q_2 \ldots q_n)$ is defined as

$$s_{\mathbf{q}} = \sup_{\mathbf{x},j} \left|\mathbf{q}(\mathbf{x}) - \mathbf{q}\left(\mathbf{x}^{(j)}\right)\right| = \max_{j \in \{1,\ldots,n\}} |q_j|,$$

where $\mathbf{x}$ ranges over all possible databases and j ranges from 1 to n.

We let $Lap(b)$ denote the one-dimensional Laplacian distribution centered at 0 with scale b and the corresponding probability density function $g(x) = \frac{1}{2 \cdot b} e^{-\frac{|x|}{b}}$.

Definition 4.5. The *Laplacian mechanism*, denoted $\mathcal{L}$, is a differentially private mechanism defined as follows: for a given query $\mathbf{Q} = (\mathbf{q}, v)$ and database instance $\mathbf{x}$, the mechanism returns $\mathcal{L}_{\mathbf{Q}}(\mathbf{x}) = \mathbf{q}(\mathbf{x}) + \rho$, where ρ is noise with distribution $Lap(b)$ and $b = \sqrt{v/2}$.

The following is known from the work on differential privacy [Dwork et al. 2006].

PROPOSITION 4.6. *Let $\mathcal{L}$ be the Laplacian mechanism, $\mathbf{Q} = (\mathbf{q}, v)$ a query, and $s_{\mathbf{q}}$ the sensitivity of* $\mathbf{q}$. *Then, the privacy loss of each individual is bounded by*

$$\varepsilon(\mathcal{L}_{\mathbf{Q}}) \leq \frac{s_{\mathbf{q}}}{\sqrt{v/2}}.$$

5. BALANCED PRICING FRAMEWORKS

In this section we define formally when a pricing framework is *balanced* and we provide a general procedure for designing a balanced pricing framework. The concept of balance brings together the three components in Figure 1: the query price π, the privacy loss ε, and the micropayments μ_i. We begin with a description of micropayments.

5.1. Micropayments to Data Owners

By answering a buyer's query $\mathbf{Q}$, using some mechanism $\mathcal{K}_{\mathbf{Q}}$, the market maker leaks some of the private data of the data owners and must compensate each data owner with some micropayment. Recall that $\mu_i(\mathbf{Q})$ denotes the sum of all micropayments due

to data owner whose attributes match x_i, the i'th entry in the data vector. The micropayments close the loop in Figure 1; they must be covered by the buyer's payment π, and must also be a function of the degree of the privacy loss ε. Micropayments must satisfy the following basic property.

Definition 5.1. Let μ_i be a micropayment function. μ_i is *micro-arbitrage free* if, for each i, $\mu_i(\mathbf{Q})$ is an arbitrage-free pricing function.

Micro-arbitrage freeness is a promise that the owner's loss of privacy will be compensated, and that there is no way for the buyer to circumvent the due micropayment by asking other queries and combining their answers. The definition is identical to that of arbitrage freeness of π.

5.2. Balanced Pricing Frameworks: Definition

The contract between data owner i and the market maker consists of a nondecreasing function $W_i : \bar{\mathbb{R}}^+ \to \bar{\mathbb{R}}^+$ such that $W_i(0) = 0$, where $\bar{\mathbb{R}}^+ = \mathbb{R}^+ \cup \{+\infty\}$. This function represents a guarantee to the data owners contributing to the i's bucket x_i will be compensated with at least $\mu_i \geq W_i(\varepsilon)$ in the event of a privacy loss ε. We denote $\mathbf{W} = (W_1, \ldots, W_n)$ the set of contracts between the market maker and all data owners.

The connection between the micropayments μ_i, the query price π, and the privacy loss ε_i is captured by the following definition.

Definition 5.2. We say that the micropayment functions μ_i, $i = 1, \ldots, n$ are *cost recovering* for a pricing function π if, for any query $\mathbf{Q}$, $\pi(\mathbf{Q}) \geq \sum_i \mu_i(\mathbf{Q})$.

Fix a query answering mechanism $\mathcal{K}$. We say that a micropayment function μ_i is *compensating* for a contract function W_i if, for any query $\mathbf{Q}$, $\mu_i(\mathbf{Q}) \geq W_i(\varepsilon(\mathcal{K}_\mathbf{Q}))$.

The market maker will insist that the micropayment functions are cost recovering; otherwise, he will not be able to pay the data owners from the buyer's payment. A data owner will insist that the micropayment function is compensating, and this enforces the contract between she and the market maker, guaranteeing she will be compensated at least $W_i(\varepsilon)$ in the event of a privacy loss ε.

Fix a query answering mechanism $\mathcal{K}$. We denote a pricing framework $(\pi, \varepsilon, \mu, \mathbf{W})$, where $\pi(\mathbf{Q})$, $\mu_i(\mathbf{Q})$ are the buyer's price and the micropayments, $\varepsilon = (\varepsilon_1, \ldots, \varepsilon_n)$, where $\varepsilon_i(\mathcal{K}_\mathbf{Q})$ is the privacy loss corresponding to the mechanism $\mathcal{K}$ and $W_i(\varepsilon)$ is the contract with the data owner i.

Definition 5.3. A pricing framework $(\pi, \varepsilon, \mu, \mathbf{W})$ is *balanced* if: (1) π is arbitrage free and (2) the micropayment functions μ are micro-arbitrage free, cost recovering for π, and compensating for $\mathbf{W}$.

We explain how the contract between the data owner and the market maker differs from that in privacy-preserving mechanisms. Let $\varepsilon > 0$ be a small constant. A mechanism $\mathcal{K}$ is called *differentially private* [Dwork et al. 2006] if, for any measurable set S, any database vector $\mathbf{x}$, and any entry j of $\mathbf{x}$,

$$\Pr(\mathcal{K}(\mathbf{x}) \in S) \leq e^\varepsilon \times \Pr\left(\mathcal{K}(\mathbf{x}^{(j)}) \in S\right).$$

In differential privacy, the basic contract between the mechanism and the data owner is the promise to every user that her privacy loss is no larger than ε. In our framework for pricing private data, we turn this contract around; now privacy *is* lost, and Definition 4.1 quantifies this loss. The contract is that the users are compensated according to their privacy loss. At an extreme, if the mechanism is ε-differentially private for a tiny ε, then each user will receive only a tiny micropayment $W_i(\varepsilon)$; as her privacy loss increases, she will be compensated more.

The micropayments circumvent a common limitation of differentially private mechanisms. In differential privacy, the data analyst typically has a fixed budget ε granted by the data curator for all queries that he may ever ask. In order to issue N queries, he needs to divide the privacy budget among these queries and, as a result, each query will be perturbed with a higher noise; after issuing these N queries, he can no longer query the database because otherwise, the contract with the data owner would be breached.

In our pricing framework there is no such hard limitation because the buyer simply pays for each query. The budget is now a real financial budget, and the buyer can ask as many queries as he wants, with as high accuracy as he wants, as long as he has money to pay for them. As a consequence, it is the analyst buyer, rather than the data owner, who ultimately determines the actual privacy loss. A similar model was proposed by Ghosh and Roth [2011] in which the ε budget is determined by the budget of the data analyst.

5.3. Balanced Pricing Frameworks: Synthesis

Call $(\varepsilon, \mu, \mathbf{W})$ *semibalanced* if all micropayment functions are micro-arbitrage free and compensating with respect to $\mathcal{K}$; that is, we leave out the pricing function π and the cost recovering requirement. The first step is to design a semibalanced set of micropayment functions.

PROPOSITION 5.4. *Let $\mathcal{L}$ be the Laplacian mechanism and let the contract functions be linear, $W_i(\varepsilon) = c_i \cdot \varepsilon$, where $c_i > 0$ is a fixed constant, for $i = 1, \ldots, n$. Define the micropayment functions $\mu_i(\mathbf{Q}) = \frac{\max_j |q_j| \cdot c_i}{\sqrt{v/2}}$, for $i = 1, \ldots, n$. Then $(\varepsilon, \mu, \mathbf{W})$ is semibalanced.*

PROOF. By Eq. (3), the function $\pi_i(\mathbf{Q}) = \frac{2(\max_j |q_j|)^2 \cdot c_i^2}{v}$ is arbitrage free. By Corollary 3.13, the function $\mu_i(\mathbf{Q}) = (\pi_i(\mathbf{Q}))^{1/2}$ is also arbitrage free, which means that μ_i is micro-arbitrage free. Finally, by Proposition 4.6, we have $W_i(\varepsilon(\mathcal{L}_\mathbf{Q})) = c_i \cdot \varepsilon(\mathcal{L}_\mathbf{Q}) \leq c_i \frac{s_q}{\sqrt{v/2}} = c_i \frac{\max_j |q_j|}{\sqrt{v/2}} = \mu_i(\mathbf{Q})$, proving that μ_i is compensating. □

Next, we show how to derive new semibalanced micropayments from existing ones.

PROPOSITION 5.5. *Suppose that $(\varepsilon, \mu^j, \mathbf{W}^j)$ is semibalanced, for $j = 1, \ldots, k$ (where $\mu^j = (\mu_1^j, \ldots, \mu_n^j)$, and $\mathbf{W}^j = (W_1^j, \ldots, W_n^j)$, for $j = 1, \ldots, k$), and let $f_i : (\bar{\mathbb{R}}^+)^k \to \bar{\mathbb{R}}^+$, $i = 1, \ldots, n$, be n nondecreasing, subadditive functions such that $f_i(\mathbf{0}) = 0$, for all $i = 1, \ldots, n$. Define $\mu_i = f_i(\mu_i^1, \ldots, \mu_i^k)$ and $W_i = f_i(W_i^1, \ldots, W_i^k)$ for each $i = 1, \ldots, n$. Then $(\varepsilon, \mu, \mathbf{W})$ is also semibalanced, where $\mu = (\mu_1, \ldots, \mu_n)$ and $\mathbf{W} = (W_1, \ldots, W_n)$.*

PROOF. By Proposition 3.12, each μ_i is arbitrage free. Finally, each μ_i is compensating for W_i because the functions f_i are nondecreasing and each μ_i^j is compensating for W_i^j, hence $f_i(\mu_i^1(\mathbf{Q}), \ldots, \mu_i^k(\mathbf{Q}) \geq f_i(W_i^1(\varepsilon(\mathcal{K}_\mathbf{Q})), \ldots, W_i^k(\varepsilon(\mathcal{K}_\mathbf{Q}))) = W_i(\varepsilon(\mathcal{K}_\mathbf{Q}))$. □

We can use this proposition to design micropayment functions that allow the true private data of an individual to be disclosed, as in Section 3.5. We illustrate with an example.

Example 5.6. Consider Example 2.2, where several voters give a rating in $\{0, 1, 2, 3, 4, 5\}$ to each of two candidates A and B. Thus x_1, x_2 represent the ratings of voter 1, x_3, x_4 of voter 2, etc. Suppose voter 1 values her privacy highly and would never accept a total disclosure: we choose linear contract functions $W_1(\varepsilon) = W_2(\varepsilon) = c \cdot \varepsilon$ for her two votes respectively, and define the micropayments as in Proposition 5.4, $\mu_i(\mathbf{Q}) = \frac{\max_j |q_j| c}{\sqrt{v/2}}$ for $i = 1, 2$. On the other hand, voter 2 is less concerned about her privacy and willing to sell the true values of each of her votes at some high price

$d > 0$: then we choose bounded contract functions $W_3(\varepsilon) = W_4(\varepsilon) = 2 \cdot d/\Pi \cdot \mathtt{atan}(\varepsilon)$ (which is subadditive, by Corollary 3.16) and define the micropayments accordingly, $\mu_i(\mathbf{Q}) = 2 \cdot d/\Pi \cdot \mathtt{atan}(\frac{\max_j |q_j|}{\sqrt{v/2}})$, for $i = 3, 4$. By Proposition 5.5 this function is also compensating and micro-arbitrage free and, moreover, it is bounded by $\mu_i \leq d$, where the upper bound d is reached by the total disclosure query ($v = 0$).

Finally, we choose a payment function so as to ensure the micropayments are cost recovering.

PROPOSITION 5.7. *(1) Suppose that* $(\varepsilon, \mu, \mathbf{W})$ *is semibalanced and define* $\pi(\mathbf{Q}) = \sum_i \mu_i(\mathbf{Q})$. *Then* $(\pi, \varepsilon, \mu, \mathbf{W})$ *is balanced.*

(2) Suppose that $(\pi, \varepsilon, \mu, \mathbf{W})$ *is balanced and* $\pi' \geq \pi$ *is any arbitrage-free pricing function. Then* $(\pi', \varepsilon, \mu, \mathbf{W})$ *is also balanced.*

PROOF. Claim (1) follows from Theorem 3.13 (the sum of arbitrage-free functions is also arbitrage free), while claim (2) is straighforward. □

To summarize, the synthesis procedure for a pricing framework proceeds as follows. Start with the simple micropayment functions given by Proposition 5.4 that ensure linear compensation for each user. Next, modify both the micropayment and the contract functions using Proposition 5.5, as desired, in order to adjust to the preferences of individual users, for example, in order to allow a user to set a price for her true data. Finally, define the query price to be the sum of all micropayments (Proposition 5.7), then increase this price freely by using any method in Corollary 3.13.

6. DISCUSSION

In this section, we discuss two problems in pricing private data and show how they affect our pricing framework. The first is how to incentivize the data owner to participate in the database and truthfully report her privacy valuations, which is reflected in her contract function W_i. This property is called *truthfulness* in mechanism design. The second concerns the protection of the privacy valuations itself, meaning that the contract W_i may also leak information to the buyer.

6.1. Truthfulness

How can we incentivize a user to participate and to reveal her true assessment for the privacy loss of a data item x_i? All things being equal, the data owner will quote an impossibly high price for even a tiny loss of her privacy. In other words, she would choose a contract function $W(\varepsilon)$ as close to ∞ as possible.

Incentivizing users to report their true valuation is a goal of *mechanism design*. This has been studied for private data only in the restricted case of a single query, and has been shown a difficult task. Ghosh and Roth [2011] show that if the privacy valuations are sensitive, then it is impossible to design truthful and individually rational direct revelation mechanisms. Fleischer and Lyu [2012] circumvent this impossibility result by assuming the privacy valuation is drawn from known probability distributions. Also, according to some experimental studies [Acquisti et al. 2009], the owner's valuation is often complicated and difficult for the owner to articulate and different people may have quite different valuations. Indeed, without a context or reference, it is hard for people to understand the valuation of their private data. The design of a truthful and private mechanism for private data, even for a single query, remains an active research topic.

We propose a simpler approach adopted directly from that introduced by Aperjis and Huberman [2012]. Instead of asking for their valuations, users are given a fixed number of options. For example, the users may be offered a choice between two contract

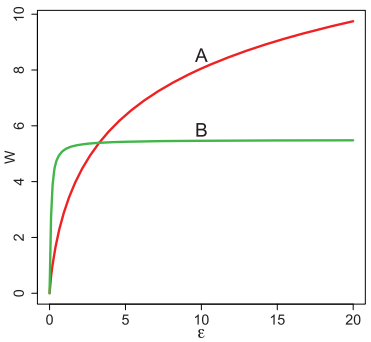

Fig. 2. Two options for the contract function W. Option A makes a small micropayment for small privacy losses and a large one for large privacy losses. Option B pays even for small privacy losses but, for large privacy losses, pays less than A. Risk-neutral users would typically choose Option A, while risk-averse ones choose Option B.

functions, shown in Figure 2, that we call Options A and B (following Aperjis and Huberman [2012]).

Option A. For a small privacy loss, the amount of micropayment is very small; for almost total privacy loss, the payment is significant.
Option B. The amount of micropayment is moderate, but almost constant in the privacy loss.

While these options were initially designed for a sampling-based query answering mechanism [Aperjis and Huberman 2012], they also work for our perturbation-based mechanism. Risk-tolerant users will typically choose Option A, while risk-averse users will choose Option B. Clearly, a good user interface will offer more than two options; designing a set of options that users can easily understand is a difficult task that we leave to future work.

6.2. Private Valuations

When users have sufficient freedom to choose their privacy valuation (i.e., their contract function W_i), then we may face another difficult problem: the privacy valuation may be strongly correlated with the data x_i itself. In this case, even releasing the price of a query may lead to privacy loss, a factor not considered in our framework. For example, consider a database of HIV status: value 1 corresponds to the data owner having HIV, while value 0 means that she does not. Typically, users who have HIV will set a much higher value on privacy of their value 1 than those who don't have HIV. Then, a savvy buyer may simply ask for the price of a query without actually purchasing it, and determine with some reasonable confidence the number of users with HIV. The problem of hiding the valuation itself is a difficult one still being actively researched in mechanism design [Ghosh and Roth 2011; Fleischer and Lyu 2012]. Instead, we describe here a simple approach based on perturbing the price itself in the same way in which we perturb the data.

More precisely, we will assume that the contract functions are also linear $W_i = c_i \cdot \varepsilon$, and therefore the price π is a linear function of the query $\mathbf{Q}$. Then, we adopt our techniques for perturbing the query answer to perturb the price itself. In other words, both $\pi(\mathbf{Q})$ and $\mu_i(\mathbf{Q})$ are perturbed in the same fashion as query answers and therefore are random variables. All three properties of arbitrage freeness, cost recovery, and compensation are now defined in terms of expected values; for example, a randomized pricing function $\pi(\mathbf{Q})$ is arbitrage free if $\{\mathbf{Q}_1, \ldots, \mathbf{Q}_m\} \to \mathbf{Q}$ implies $\mathbf{E}(\pi(\mathbf{Q})) \leq \sum_{i=1}^{m} \mathbf{E}(\pi(\mathbf{Q}_i))$,

and the micropayments are cost recovering if $\mathbf{E}(\pi(\mathbf{Q})) \geq \sum_i \mathbf{E}(\mu_i(\mathbf{Q}))$. In this setting, queries are answered using a mechanism $\mathcal{K}$, while prices are computed using a (possibly different) mechanism $\mathcal{K}'$.

The privacy loss for data item x_i includes two parts. One part is due to the release of the query answer, and the other to the release of the price. Their values are $\varepsilon(\mathcal{K})$ and $\varepsilon(\mathcal{K}')$, respectively. A micropayment is *compensating* if $\mathbf{E}(\mu_i(\mathbf{Q})) \geq c_i \cdot (\varepsilon(\mathcal{K}) + \varepsilon(\mathcal{K}'))$.

We describe here a balanced mechanism where data owners are compensated both for the privacy loss of their data *and* for the privacy loss of the price of their data. The idea is very simple. When the buyer asks for the price of the query $\pi(\mathbf{Q})$, which is now a perturbed quantity, the price itself leaks a bit of privacy ε_1. As a consequence, the buyer needs to pay an incremental price π_1 (which is $\sum_i c_i * \varepsilon_1$). Once informed about this new price, there is another leak of privacy ε_2. The buyer needs to pay another incremental price π_2. This in turn results in another leak of privacy ε_3, etc. The sum of all these prices is a geometric series. When it converges, then it represents a price that pays not only for the leak of the data, but also (reflexively) for the leak of the price itself. In the rest of this section we give the technical details for this idea.

As for the data items, we assume that the constants c_i used in the contract function are drawn from a bounded domain $Y \subseteq \mathbb{R}$, and denote $\delta = \sup_{c \in Y} |c|$. Assume both $\mathcal{K}$ and $\mathcal{K}'$ are Laplacian mechanisms. Given a query $\mathbf{Q} = (\mathbf{q}, v)$, set $b = \sqrt{v/2}$, choose some[8] $b' > \delta$, tunable by the market maker. $\mathcal{K}$ is the mechanism that on an input $\mathbf{x}$ returns $\mathbf{q}(\mathbf{x}) + \rho$, where ρ is a noise with distribution $Lap(b)$. $\mathcal{K}'$ is the mechanism that on an input $\mathbf{c}$ returns a noisy price $\frac{b' \max_j |q_j|}{b \cdot (b' - \delta)} \sum_i c_i + \rho'$, where ρ' is a noise with distribution $Lap(b')$. We denote the exact price $\frac{b' \max_j |q_j|}{b \cdot (b' - \delta)} \sum_i c_i$ as $\mathbf{E}(\mathcal{K}'(\mathbf{c}))$. The sensitivity of the mechanism $\mathcal{K}$ is $s_i(\mathcal{K}) = \max_j |q_j|$ (Definition 4.4). If we define $s_i(\mathcal{K}') = \frac{b' \delta \max_j |q_j|}{b \cdot (b' - \delta)}$, then we have

$$\varepsilon(\mathcal{K}) \leq \frac{s_i(\mathcal{K})}{b}, \quad \varepsilon(\mathcal{K}') \leq \frac{s_i(\mathcal{K}')}{b'}.$$

PROPOSITION 6.1. *Let $\mathcal{K}, \mathcal{K}'$ be Laplacian mechanisms (as described before) and $\mathbf{Q} = (\mathbf{q}, v)$ a query. Set (as earlier) $b = \sqrt{v/2}$ and $b' > \delta$. Define*

$$\begin{aligned} \pi(\mathbf{Q}) &= \mathcal{K}'(\mathbf{c}) = \mathbf{E}\left(\mathcal{K}'(\mathbf{c})\right) + \rho' \\ \mu_i(\mathbf{Q}) &= \left(\frac{s_i(\mathcal{K})}{b} + \frac{s_i(\mathcal{K}')}{b'}\right) \cdot c_i + \frac{\pi(\mathbf{Q}) - \mathbf{E}\left(\mathcal{K}'(\mathbf{c})\right)}{n}, \\ & \quad \forall i = 1, \ldots, n. \end{aligned}$$

Then $(\pi, \mu, \varepsilon, \mathbf{W})$ is a balanced mechanism.

PROOF. We show that μ_i is micro-arbitrage free in expectation. For each individual i, by definition,

$$\begin{aligned} \mathbf{E}(\mu_i(\mathbf{Q})) &= \frac{b' \cdot c_i \cdot \max_j |q_j|}{b \cdot (b' - \delta)} \\ &= \frac{\sqrt{2} b' \cdot c_i}{b' - \delta} \frac{\max_j |q_j|}{\sqrt{v}}. \end{aligned}$$

By the same argument as in Proposition 5.4, $\mathbf{E}\left(\mu_i(\mathbf{Q})\right)$ is arbitrage free and thus $\mu_i(\mathbf{Q})$ is arbitrage free in expectation.

[8]When $b' \leq \delta$, the expectation of the price π is infinite.

We show that the micropayments are cost recovering. By definition,

$$\begin{aligned}\sum_i \mu_i(\mathbf{Q}) &= \sum_i \left(\frac{s_i(\mathcal{K})}{b} + \frac{s_i(\mathcal{K}')}{b'} \right) \times c_i + \rho' \\ &= \sum_i \left(\frac{\max_j |q_j|}{b} + \frac{\frac{b'\delta \max_j |q_j|}{b\cdot(b'-\delta)})}{b'} \right) \times c_i + \rho' \\ &= \frac{b' \max_j |q_j|}{b \cdot (b' - \delta)} \sum_i c_i + \rho' \\ &= \pi(\mathbf{Q}),\end{aligned}$$

proving the claim.

Finally, we show that μ_i is compensating, in expectation. For each individual i,

$$\begin{aligned}\mathbf{E}(\mu_i(\mathbf{Q})) &= \left(\frac{s_i(\mathcal{K})}{b} + \frac{s_i(\mathcal{K}')}{b'} \right) \times c_i \\ &\geq (\varepsilon(\mathcal{K}) + \varepsilon(\mathcal{K}') \times c_i,\end{aligned}$$

meaning that $\mu_i(\mathbf{Q})$ compensates user i for her loss of privacy in expectation.

By a similar argument as in Proposition 5.7, $\pi(\mathbf{Q})$ is arbitrage free in expectation. □

7. RELATED WORK

Recent investigation of the trade-off between privacy and utility in statistical databases was initiated by Dinur and Nissim [2003] and culminated in Dwork et al. [2006], where the authors introduced *differential privacy* and the *Laplace mechanism*. The goal of this line of research is to reveal accurate statistics while preserving the privacy of the individuals. There have been two (somewhat artificially divided) models involved: the noninteractive model and the interactive model. In this article, we use an interactive model in which queries arrive online, one at a time, and the market maker has to charge for them appropriately and answer them. There is a large and growing literature on differential privacy; we refer readers to the recent survey by Dwork [2011]. There is privacy loss in releasing statistics in a differentially private sense (quantified in terms of the privacy parameter/budget ε). However, this line of research does not consider compensating the privacy loss.

Ghosh and Roth [2011] initiated a study of how to incentivize individuals to contribute their private data and to truthfully report their privacy valuation using tools of mechanism design. They consider the same problem as we do namely pricing private data, but from a different perspective where there is only one query and the individuals' valuations of their data are private. The goal is to design a truthful mechanism for disclosing the valuation. In contrast, we assume the individuals' valuations are public, and focus instead on the issues arising from pricing multiple queries consistently. Another key difference is that we require not only accuracy but also unbiasedness for the noisy answer to a certain query, while in Ghosh and Roth [2011] answers are not unbiased. There have been some follow-ups to Ghosh and Roth [2011], such as Fleischer and Lyu [2012], Ligett and Roth [2012], Roth and Schoenebeck [2012], and Dandekar et al. [2011]; a good survey is Roth [2012]. There are also other papers that consider privacy and utility in the context of mechanism design, such as Nissim et al. [2012] and Chen et al. [2011].

Economic perspectives on the regulation and control of private information have a long history [Stigler 1980; Posner 1981]. A national information market where personal information could be bought and sold was proposed by Laudon [1996]. Garfinkel et al.

[2006] proposed a methodology for releasing approximate answers to statistical queries and compensating contributing individuals as the basis for a market for private data. That methodology does not use a rigorous measure of privacy loss or protection and does not address the problem of arbitrage.

Recently, Balazinska et al. [2011] initiated a study of data markets in the cloud (for general-purpose data, not specifically private data). Subsequently, Koutris et al. [2012] proposed a data pricing method that first sets explicit price points on a set of views and then computes the implied price for any query. However, they did not consider the potential privacy risks of their method. The query determinacy used in Koutris et al. [2012] is instance based and, as a result, the adversary could (in the worst case) learn the entire database solely by asking the prices of queries (for free). Li and Miklau [2012] study data pricing for linear aggregation queries using a notion of instance-independent query determinacy. This avoids some privacy risks, but it is still sometimes possible to infer query answers for which the buyer has not paid. Both of the prior works consider a model in which unperturbed query answers are exchanged for payment. In this article we consider noisy query answers and use an instance-independent notion of query determinacy that allows us to formally model private disclosures and assign prices accordingly.

Aperjis and Huberman [2012] describe a simple strategy to collect private data from individuals and compensate them based on an assumption in sociology that some people are risk averse. By doing so, buyers could compensate individuals with relatively less money. More specifically, a buyer may access the private data of an individual with probability 0.2 and offer her two choices: if the data is accessed, then she would be paid \$10, otherwise she would receive nothing; she would receive \$1 regardless of whether her data would be used or not. Then a risk-averse person may choose the second choice and consequently the buyer can save \$1 in expectation. In their paper, the private data of an individual is either entirely exposed or completely unused. In our framework, there are different levels of privacy, the privacy loss is carefully quantified and compensated, and thus the data is better protected. Finally, Riederer et al. [2011] propose auction methods to sell private data to aggregators, but an owner's data is either completely hidden or totally disclosed and the price of data is ultimately determined by buyers without consideration of owners' personal privacy valuations.

The current article is based on our preliminary work [Li et al. 2013]. There, we introduced the idea of using perturbation to reduce the price of private data sold to a data analyst, but we used a simpler data model where each record corresponds to one user; in the current article we replaced it with the standard vector representation used in the literature, where each record corresponds to one combination of attributes plus the number of users having these attributes. As a result, the notion of privacy for individuals is now different. An important question that we did not address in Li et al. [2013] is, given a set of view-price pairs ("price points"), can a particular query be answered within a given budget by purchasing, possibly repeatedly, the views in the set? This question now treated in Section 3.6. Several proofs omitted in Li et al. [2013] are also included here.

8. CONCLUSIONS

We have introduced a framework for selling private data. Buyers can purchase any linear query, with any amount of perturbation, and need to pay accordingly. Data owners, in turn, are compensated according to the privacy loss they incur for each query. In our framework buyers are allowed to ask an arbitrary number of queries, and we have designed techniques for ensuring that the prices are arbitrage free according to a specific definition of arbitrage freeness, meaning that buyers are guaranteed to pay for any information that they may further extract from the queries. Our pricing

framework is balanced in the sense that the buyer's price covers the micropayments to the data owner and each micropayment compensates the users according to their privacy loss.

An interesting open question is whether we can achieve both truthfulness (as discussed in Ghosh and Roth [2011]) and arbitrage freeness (as discussed in the current work) when pricing private data. Further, it remains to consider general notions of answerability that go beyond linear answerability, or to bound the impact that nonlinear estimation methods could have in the context of arbitrage.

REFERENCES

Alessandro Acquisti, Leslie John, and George Loewenstein. 2009. What is privacy worth? In *Proceedings of the Workshop on Information Systems and Economics (WISE'09)*.

Christina Aperjis and Bernardo A. Huberman. 2012. A market for unbiased private data: Paying individuals according to their privacy attitudes. *First Monday* 17, 5.

Magdalena Balazinska, Bill Howe, and Dan Suciu. 2011. Data markets in the cloud: An opportunity for the database community. *Proc. VLDB Endow.* 4, 12, 1482–1485.

Avrim Blum. 2003. Machine learning: My favorite results, directions, and open problems. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS'03)*. 2.

Stephen P. Boyd and Lieven Vandenberghe. 2004. *Convex Optimization*. Cambridge University Press.

Joshua Brustein. 2012. Start-ups seek to help users put a price on their personal data. *The New York Times* (Feb. 2012)

Bee-Chung Chen, Daniel Kifer, Kristen Lefevre, and Ashwin Machanavajjhala. 2010. Privacy-preserving data publishing. *Foundat. Trends Databases* 2, 1–2, 1–167.

Yiling Chen, Stephen Chong, Ian A. Kash, Tal Moran, and Salil P. Vadhan. 2011. Truthful mechanisms for agents that value privacy. In *Proceedings of the 14th ACM Conference on Electronic Commerce (EC'11)*. 215–232.

Pranav Dandekar, Nadia Fawaz, and Stratis Ioannidis. 2011. Privacy auctions for inner product disclosures. http://arxiv.org/abs/arXiv:1111.2885.

George Danezis and Seda Gurses. 2010. A critical review of 10 years of privacy technology. In *Proceedings of the Surveillance Cultures Conference: A Global Surveillance Society*.

Irit Dinur and Kobbi Nissim. 2003. Revealing information while preserving privacy. In *Proceedings of the 22nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS'03)*. 202–210.

Cynthia Dwork. 2011. A firm foundation for private data analysis. *Comm. ACM* 54, 1, 86–95.

Cynthia Dwork, Frank Mcsherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the 3rd Conference on Theory of Cryptography (TCC'06)*. 265–284.

Cynthia Dwork and Sergey Yekhanin. 2008. New efficient attacks on statistical disclosure control mechanisms. In *Proceedings of the 28th Annual Conference on Cryptology: Advances in Cryptology (CRYPTO'08)*. 469–480.

Lisa Fleischer and Yu-Han Lyu. 2012. Approximately optimal auctions for selling privacy when costs are correlated with data. In *Proceedings of the ACM Conference on Electronic Commerce (EC'12)*. 568–585.

Benjamin C. M. Fung, Ke Wang, Rui Chen, and Philip S. Yu. 2010. Privacy-preserving data publishing: A survey of recent developments. *ACM Comput. Surv.* 42, 4.

Michael R. Garey and David S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman.

Robert S. Garfinkel, Ram D. Gopal, Manuel A. Nunez, and Daniel Rice. 2006. Secure electronic markets for private information. *IEEE Trans. Syst. Man, Cybernet.* A36, 3, 461–471.

Arpita Ghosh and Aaron Roth. 2011. Selling privacy at auction. In *Proceedings of the ACM Conference on Electronic Commerce (EC'11)*. 199–208.

Alon Y. Halevy. 2001. Answering queries using views: A survey. *VLDB J.* 10, 4, 70–294.

Moritz Hardt, Katrina Ligett, and Frank Mcsherry. 2012. A simple and practical algorithm for differentially private data release. In *Proceedings of the Conference on Advances in Neural Information Processing Systems (NIPS'12)*. 2348–2356.

Moritz Hardt and Guy N. Rothblum. 2010. A multiplicative weights mechanism for privacy-preserving data analysis. In *Proceedings of the 51st IEEE Annual Symposium on Foundations of Computer Science (FOCS'10)*. 1–70.

Daniel Kifer, John Abowd, Johannes Gehrke, and Lars Vilhuber. 2008. Privacy: Theory meets practice on the map. In *Proceedings of the 24th IEEE International Conference on Data Engineering (ICDE'08)*. 277–286.

Henning Knautz. 1999. Nonlinear unbiased estimation in the linear regression model with nonnormal disturbances. *J. Statist. Plan. Infer.* 81, 2, 293–309.

Paraschos Koutris, Prasang Upadhyaya, Magdalena Balazinska, Bill Howe, and Dan Suciu. 2012. Query-based data pricing. In *Proceedings of the 31st Symposium on Principles of Database Systems (PODS'12)*. 167–178.

Kenneth C. Laudon. 1996. Markets and privacy. *Comm. ACM* 39, 9, 92–104.

Chao Li, Michael Hay, Vibhor Rastogi, Gerome Miklau, and Andrew Mcgregor. 2010. Optimizing linear counting queries under differential privacy. In *Proceedings of the 29th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS'10)*. 123–134.

Chao Li, Daniel Yang Li, Gerome Miklau, and Dan Suciu. 2013. A theory of pricing private data. In *Proceedings of the 16th International Conference on Database Theory (ICDT'13)*. 33–44.

Chao Li and Gerome Miklau. 2012. Pricing aggregate queries in a data marketplace. In *Proceedings of the 15th International Workshop on the Web and Databases (WebDB'12)*. 19–24.

Katrina Ligett and Aaron Roth. 2012. Take it or leave it: Running a survey when privacy comes at a cost. In *Proceedings of the 8th International Conference on Internet and Network Economics (WINE'12)*. 378–391.

Frank Mcsherry. 2010. Privacy integrated queries: An extensible platform for privacy-preserving data analysis. *Comm. ACM* 53, 9, 89–97.

Alan Nash, Luc Segoufin, and Victor Vianu. 2010. Views and queries: Determinacy and rewriting. *ACM Trans. Database Syst.* 35, 3.

Kobbi Nissim, Claudio Orlandi, and Rann Smorodinsky. 2012. Privacy-aware mechanism design. In *Proceedings of the ACM Conference on Electronic Commerce (EC'12)*. 774–789.

Richard A. Posner. 1981. The economics of privacy. *Amer. Econ. Rev.* 71, 2, 405–409.

Christopher Riederer, Vijay Erramilli, Augistin Chaintreau, Balachander Krishnamurthy, and Pablo Rodriguez. 2011. For sale: Your data: By: You. In *Proceedings of the 10th ACM Workshop on Hot Topics in Networks (HotNets'11)*.

Aaron Roth. 2012. Buying private data at auction: The sensitive surveyor's problem. *SIGecom Exch.* 11, 1, 1–8.

Aaron Roth and Grant Schoenebeck. 2012. Conducting truthful surveys, cheaply. In *Proceedings of the ACM Conference on Electronic Commerce (EC'12)*. 826–843.

George J. Stigler. 1980. An introduction to privacy in economics and politics. *J. Legal Stud.* 9, 4, 623–644.

World Economic Forum. 2011. Personal data: The emergence of a new asset class. Report of the world economic forum. http://www.weforum.org/reports/personal-data-emergence-new-asset-class.

Ganzhao Yuan, Zhenjie Zhang, Marianne Winslett, Xiaokui Xiao, Yin Yang, and Zhifeng Hao. 2012. Low-rank mechanism: Optimizing batch queries under differential privacy. *Proc. VLDB Endow.* 5, 11, 1352–1363.

Received October 2013; revised August 2014; accepted August 2014