A Discussion on Pricing Relational Data

Magdalena Balazinska, Bill Howe, Paraschos Koutris,
Dan Suciu, and Prasang Upadhyaya

University of Washington,
Seattle, USA

Abstract. There exists a growing market for structured data on the Internet today, and this motivates a theoretical study of how relational data should be priced. We advocate for a framework where the seller defines a *pricing scheme*, by essentially stipulating the price of some queries, and the buyer is allowed to purchase data expressed by any query they wish: the system will derive the price automatically from the pricing scheme. We show that, in order to understand pricing, one needs to understand *determinacy* first. We also discuss some other open problems in pricing relational data.

Keywords: relational databases, pricing

1 Introduction

In the summer of 2007, Peter Buneman posed the following question to one of the authors of this article. How should one set a price for data on the Internet? A lot of data is freely available today, but for some data the production costs are quite high, and it makes sense to charge for its usage in order to recover the production costs. Peter's original motivation came from the IUPHAR database [1], a repository of receptor nomenclature and drug classifications contributed by a large community of experts in the field. Observing that this data is extremely valuable to pharmaceutical companies, Peter reasoned that one could recover some of the costs of producing and maintaining the data by charging these pharmaceutical companies a price for accessing it. Some technical developments resulting from those initial discussions with Peter are available in a separate manuscript [9].

Today, Peter's question applies to a large number of datasets, both from the scientific and commercial domains; increasingly, one finds data for sale on the Internet. In fact, in recent years, one has witnessed the emergence of *marketplace services for data*, which are Websites whose purpose is to facilitate buying and selling data. Examples of such data marketplaces are the Windows Azure Marketplace [2], a data marketplace that contains over 100 data sources for sale, Infochimps [3], which contains about 15,000 data sets for sale, and Xignite [5], which sells financial data.

The database group at the University of Washington has started a research project on data markets. Funded by a partnership between NSF and Microsoft, the project plans to investigate several aspects of data markets, ranging from

systems issues arising from monitoring data usage for billing purposes, to understanding the principles of the interaction between data and prices [4,8]. In this paper, we outline our initial investigation into the latter: how to mix data and prices in a principled way. Our thinking was, in part, informed by those early discussions with Peter in 2007.

2 Of Versions and Views

On the surface, buying and selling data is not much different from buying and selling any other products. An agent produces the data and incurs some cost in doing so; the data has some value to a buyer; the seller and buyer agree on a price. This is a problem studied extensively by economists over centuries. However, as explained by Shapiro and Varian [16], digital goods, of which data sets are one instance, have unique characteristics that cause traditional pricing mechanisms to fail: they have a high and irrecoverable fixed cost (producing the data is expensive) and a very low variable cost (copying the data is almost free). The fixed and irrecoverable cost of data is quite distinct from that of physical goods. Shapiro and Varian illustrate this with a large airplane manufacturing company investing in a new factory: if the business plan turns sour, the company can still recover some of its investment by reselling the building and the manufacturing machinery. In contrast, if a company invests in acquiring detailed satellite data, and is undercut by a competitor selling similar data at a much lower price, it cannot recover anything from its now worthless satellite data. The low cost of copying digital goods further exacerbate the problem, allowing competitors to churn out copies in unrestricted quantities. The sharp skew towards fixed costs makes traditional cost-based pricing models inapplicable. This can lead either to fortunes for the producer (if she has no competition), or to total ruin.

Shapiro and Varian [16] argue that pricing on the Internet should be based on the value that a customer places on the information. They argue that *versioning* digital products is the solution to pricing digital goods. Even pricing traditional information products included some form of versioning. In the case of movies, the "new-release" version costs $12/person to watch, but renting the "DVD" version that comes out six month later costs $3/family; the two versions target two kinds of buyers, the must-see-it-now buyers willing to pay an extra price, and the price-conscious buyers who can wait six months.

The analog to versions in data markets are *views*. A view over a data instance is the same as a version of that instance. The view may contain only a subset of the data, or only some columns, or may contain information at a coarser granularity. All these can be seen as different versions of the digital product, and sold at different prices.

Consider, for example, a dataset stored in a single relation $R(x, y, z)$. The seller could set two price levels: a price p_1 for the entire dataset, and a price p_2 for an individual tuple. Presumably, the former price is much higher than the latter, $p_1 \gg p_2$. As a concrete example, it is possible today to buy either entire databases of curated business addresses [11] or to check the correctness

of individual addresses [13]. This corresponds to two versions, one for the power customers, who need the entire dataset and are willing to pay a high price, and a second version for the occasional customer interested in only one or just a few records.

Dataset versions are commonly used today. For example, CustomLists.net [11] sells a database of 28.6 million American businesses for $399. The price is only $199 for a single state and it is only $299 for the subset of American businesses that also have an email address. Such versions add significant flexibility, but what if a user wants some other subset of the data such as only large businesses with more than 1000 employees? Or businesses within 1 mile of a Home Depot? Or businesses in cities that experience frequent flooding? Today, buyers must either purchase supersets of the data they need or they must negotiate custom data products. AggData [7] is an example data seller that provides such custom solutions. Negotiating custom solutions, however, does not scale: If a human must look at each custom view and must price that view, possibly negotiating with the buyer, the total number of distinct views that can be priced is limited.

We envision a solution that allows the seller to assign a price to any possible view that the buyer may be willing to buy. This requires a study of how database views can be adorned with prices. We start with the following definition.

Definition 1. *Let D be a database instance. A* pricing scheme *for D is a set of view, price pairs: $S = \{(V_1, p_1), \ldots, (V_k, p_k)\}$.*

The data seller decides to create k "versions" of her digital product, defined by k views, and price each of them differently. The goal is to define some high-value views (for example, the entire dataset) to be sold to a few high rollers, yet define sufficiently many lower quality views that can be sold to a large number of customers. From these k views, the goal is to *automatically* derive the price of any other view V defined by the buyer. This is also the direction in which the initial discussion with Peter was heading in 2007: set the prices of some subsets, and infer automatically the prices of all other subsets [9].

An important problem that needs to be studied in pricing data is the choice of the view language in which we express the views $V_1, \ldots, V_k$ in Definition 1. This is non-trivial: we discuss here three dimensions of this problem, leaving a solution to future work.

Relational View Any selections or projections should be available to the seller if she decides to set a price on that selection or projection. We argue that joins are needed too. For example, suppose the seller wants to set a certain price for the personal information of all CEO's of companies with a revenue $> \$10M$: this requires a semijoin of the CEO relation with the Company relation. In general, one can make the argument that the seller should be allowed to use *arbitrary relational views* to define versions of the data.

Increasing/Decreasing Accuracy Decreasing the accuracy or adding noise to the data can produce a version that is less valuable, and, hence, can be sold at a lower price, to a larger number of buyers. For example, weather data

for standard, city-wide weather forecast is virtually free, but detailed precipitation information required by commercial farmers can only be purchased at a cost. There is an interesting connection here to data privacy: private data is sold today at a price, but properly anonymized data is free. The converse is also true: by performing data cleaning, the seller may increase the value of her data product. *Views that add noise to the data* should be available to the seller to set prices.

User-defined Functions The seller may own a domain specific algorithm for enhancing the data; by applying that function, the seller can produce new data that is more valuable than the raw data. For example, the seller may have a proprietary algorithm for image processing; by applying this function to all images in a collection it may produce a more valuable data set. Another example consists of a sophisticated data mining algorithm: the result of the data mining is much more valuable that the raw data itself. The seller should be able to define *views with user-defined functions.*

3 Arbitrage in Data Pricing

Consider a pricing scheme S given by Definition 1. Two problems may arise.

The first is consistency. One expects that every price point (V_i, p_i) will make sense. For example, it does not make sense to charge more for a single tuple than for the entire dataset. In similar spirit, if the entire relation R costs p_1 and a single tuple in R costs p_2, then it does not make sense to have $|R| \cdot p_2 < p_1$, or, else, no buyer will buy the entire dataset, but would instead buy one tuple at a time. We say that a pricing scheme $S = \{(V_1, p_1), \ldots, (V_k, p_k)\}$ is *consistent* if no view V_i can be obtained at a price lower than p_i by purchasing and combining some of the other views in S. The consistency problem is this: Given a pricing scheme S, check whether it is consistent.

The second problem is pricing a new view. Continuing the example where p_1 is the price for the entire data set and p_2 is the price for each individual record, how much should a buyer pay if she wants to buy half of the data records ? On one hand she could buy the entire dataset and pay p_1, then retain only the half she needs. On the other hand, she could purchase one record at a time, and pay $|R| \cdot p_2/2$. Clearly, the buyer will choose whichever is cheaper. In general, the *price computation problem* is this: given a pricing scheme $S = \{(V_1, p_1), \ldots, (V_k, p_k)\}$ and a new view V (not necessarily mentioned in S), determine the cheapest way for a user to obtain V by purchasing views available in S.

Both problems are facets of *arbitrage*. Arbitrage occurs if the pricing scheme sets a price p for a view V (possibly a new view not explicitly priced by the seller), but a buyer has the option of answering V from $V_1, \ldots, V_m$ such that their combined price is less than p: not only can the buyer get away by paying less than p, but she could even profit by reselling V at a price lower than p, which is traditionally called *arbitrage*.

The key technical difficulty in studying arbitrage is determining when a buyer can answer a view V using the information in other views, say $V_1, \ldots V_m$. Let us

write:

$$V_1, \ldots, V_m \twoheadrightarrow V \tag{1}$$

if V can be answered from the views $V_1, \ldots, V_m$. This is just a notation, not a formal definition; the intuition is that a buyer who needs the view V would rather purchase the views $V_1, \ldots, V_m$ and compute V, if these m views are cheaper than the price of V. We will discuss later how to define $\twoheadrightarrow$. Assuming that $\twoheadrightarrow$ is given, one can define both consistency and the price function.

Definition 2. *A pricing scheme* $S = \{(V_1, p_1), \ldots, (V_k, p_k)\}$ *is* consistent *if, whenever* $V_{i_1}, V_{i_2}, \ldots, V_{i_k} \twoheadrightarrow V_i$*, then* $p_i \leq p_{i_1} + p_{i_2} + \ldots + p_{i_k}$*. Given a view* V *and a pricing scheme* $S = \{(V_1, p_1), \ldots, (V_k, p_k)\}$*, let* $S \twoheadrightarrow V$ *indicate* $\{V_1, V_2, \ldots, V_k\} \twoheadrightarrow V$*. Then the* price function defined by S *is*

$$p_S(V) = \min_{T \subseteq S, T \twoheadrightarrow V} \sum_{(V_i, p_i) \in T} p_i$$

In other words, S is consistent if a buyer cannot obtain V_i by paying less than p_i. Moreover, the price of an arbitrary view V is obtained by choosing the least expensive subset of S that can be used to answer V, where the price of $T \subseteq S$ is just the sum of the prices of the views it contains.

We can also formally express the property that a pricing function does not allow arbitrage.

Definition 3. *A pricing function* p *is* arbitrage-free *if, whenever* $V_1, \ldots, V_m \twoheadrightarrow V$*, then* $p(V) \leq \sum_{i=1}^{m} p(V_i)$*.*

Note that this definition does not assume any pricing scheme S; for example, the constant pricing function that assigns the same price to every view is arbitrage-free. Now assume that a pricing scheme S is given, and consider the pricing function p_S defined in Definition 2. We were able to prove two interesting facts (assuming some natural properties for $\twoheadrightarrow$). First, p_S is arbitrage-free; second, S is consistent iff for every price point (V_i, p_i) is S, the following holds: $p_S(V_i) = p_i$.

We end this section with a discussion on the key technical difficulty of pricing: How should we define $\twoheadrightarrow$ in Equation 1? Database theoreticians have studied *query answering using views* for almost two decades, starting with Levy [12], and Abiteboul and Duschka [6]. More recently, Segoufin and Vianu [15] and Nash, Segoufin, and Vianu [14] have revisited the notion of query answering using information-content. Their formal definition of determinacy is equivalent to the following: $V_1, \ldots, V_m \twoheadrightarrow V$ if there exists a function f such that, for any database instance D, $f(V_1(D), \ldots, V_m(D)) = V(D)$. Let us call this definition of determinacy NSV. If one adopts NSV for pricing, then, given any pricing scheme S, the equation from Definition 2 extends it uniquely to a global pricing function p_S. We argue, however, that NSV is not the right notion for defining p_S, and therefore a different definition for $\twoheadrightarrow$ is needed in order to compute prices. Specifically:

– NSV is insensitive to the data instance. That means that the pricing function $p_S(V)$ depends only on the view V, and not on the database instance D. In practice, the database instance is also a variable, and should be considered as input to the pricing function. For example the seller may add more data to her raw dataset; as a consequence, she wants her pricing function to increase. The determinacy relation $\twoheadrightarrow$ should somehow depend on the database instance too. *Instance-based* determinacy has been much less studied in the literature; one such definition can be found in Calvanese *et al.* [10].
– Unfortunately, NSV is difficult to check: it is undecidable for unions of conjunctive queries, and its decidability is open for conjunctive queries [14]. This means that we do not have any practical means for computing the pricing function $p_S(V)$.
– NSV deals incorrectly with user-defined functions. For example, consider a view $V(x, f(y, z)) = R(x, y, z)$ that applies a proprietary user-defined function f to the attributes y and z. Naturally, the seller would like to charge more for V than for R, but R determines V, because, mathematically, one can compute V from R. NSV does not capture the fact that f is a proprietary function, which cannot be applied by the user interested in computing V from R.
– Noise and levels of accuracy are not captured by NSV either, because the latter is, in essence, a deterministic definition. We are not aware of any natural extension of the determinacy relation $\twoheadrightarrow$ that can deal with noise in the data.

To summarize, in order to understand the price of data one must understand the notion of determinacy first. There exists an elegant definition for the latter, but that does not seem to be the right choice for setting prices.

4 Open Problems

Data markets motivate a new direction of research in database theory. While we have discussed the determinacy relation as the first step of this research, it is by far not the only one. Several other open problems exists, we briefly mention a few here.

Pricing updates The interaction between updates and prices is interesting. The seller expects its prices to increase once the data is updated (assuming tuples are being inserted), which seems to impose additional requirements on a pricing function. At a more practical level, one question is how to charge the buyer for incremental updates: if he already purchased data from the old version, he expects to pay a reduced price for the updates. Finally, it is unclear how consistency or arbitrage are affected by updates: if S is consistent, can it become inconsistent after an update?

Pricing integrated data The interraction between multiple vendors affects the pricing function in interesting ways. For example, different vendors may

add value in different ways to same data: the first vendor provides raw images, the second runs a proprietary face recognition algorithm, and the third integrates the extracted faces with a social network database, thus putting names on pictures. Each vendor adds some value to the data, by integrating it with her own dataset or her proprietary tools. It will be quite challenging to define pricing functions in such complex scenarios.

Pricing competing data sources There are often multiple vendors for quite similar data sources. For example, today one can buy data about businesses from several vendors. There are subtle relationships between these sources: some are more complete, others are more accurate, others are more up to date, while others yet are more reliable. Another major challenge is to understand how prices are affected by competing data sources.

References

1. `http://www.iuphar-db.org/`.
2. `https://datamarket.azure.com/`.
3. `http://www.infochimps.com/`.
4. The Data Eco$y$tem project: Data management and pricing in the cloud. `http://data-pricing.cs.washington.edu/`.
5. `http://www.xignite.com/`.
6. Abiteboul, S., and Duschka, O. M. Complexity of answering queries using materialized views. In *PODS* (1998), ACM Press, pp. 254–263.
7. `http://www.aggdata.com/`.
8. Balazinska, M., Howe, B., and Suciu, D. Data markets in the cloud: An opportunity for the database community. *Proc. of the VLDB Endowment 4*, 12 (2011).
9. Buneman, P., and Suciu, D. Censoring and pricing data. Manuscript, July 2007.
10. Calvanese, D., Giacomo, G. D., Lenzerini, M., and Vardi, M. Y. Lossless regular views. In *PODS* (2002), L. Popa, Ed., ACM, pp. 247–258.
11. `http://www.customlists.net/`.
12. Levy, A. Y., Mendelzon, A. O., Sagiv, Y., and Srivastava, D. Answering queries using views. In *PODS* (1995), pp. 95–104.
13. `https://datamarket.azure.com/dataset/59a168b8-6d66-4f85-b000-38abcad310a2`.
14. Nash, A., Segoufin, L., and Vianu, V. Determinacy and rewriting of conjunctive queries using views: A progress report. In *ICDT* (2007), T. Schwentick and D. Suciu, Eds., vol. 4353 of *Lecture Notes in Computer Science*, Springer, pp. 59–73.
15. Segoufin, L., and Vianu, V. Views and queries: determinacy and rewriting. In *PODS* (2005), C. Li, Ed., ACM, pp. 49–60.
16. Shapiro, C., and Varian, H. R. Versioning: The smart way to sell information. *Harvard Business Review 76* (November-December 1998), 106–114.