A Double Auction for Querying the Web of Data

Dmitry Moor Tobias Grubenmann Sven Seuken Abraham Bernstein
Department of Informatics
University of Zurich

July 23, 2015

Abstract

Currently, the Web of Data (WoD) suffers from a lack of financial incentives for data providers. In this paper, we address this issue, by proposing a double auction to efficiently allocate answers (from data providers) to queries in the WoD. However, our domain exhibits a number of complicating features. Most importantly, before executing a particular query, the market mechanism only has estimates regarding what result can be expected. Thus, in contrast to other domains, the allocation rule as well as the pricing rule of the auction must operate based on *value* and *cost estimates*. New challenges arise from this setting; in particular the auction's *participation* constraint can no longer be guaranteed to be satisfied. We propose three payment correction rules to address this issue, and compare the efficiency of the resulting payment rules via a *computational Bayes-Nash equilibrium* analysis.

1 Introduction

The *Web of Data (WoD)* is to computers what the traditional Web is to humans. The goal is to expose data in a semantic format such that machines can easily find the information they are looking for. This semantic format allows for easy data integration and hence content from different sources can be queried in a federated fashion without the need to agree on a common scheme – something that is neither possible in the traditional Web nor in a traditional database setting. If implemented properly, a *semantic search* would lead to the desired results much faster than using a traditional search engine. However, a big practical challenge is the adoption of WoD technologies (Antoniou and van Harmelen, 2004). In particular, the majority of data that exists somewhere in some form is simply not made available in a WoD format.

The primary reason for this is a lack of incentives for the data providers: on the WoD, datasets are usually queried by algorithms rather than viewed by people. Thus, *advertising*, the main source of income for search engines on the traditional web, does not work in this environment where machines process the data and automatically filter out unwanted information or advertisement.

It is of course possible to directly charge users to access (and search through) high-quality data. For example, *Bloomberg*, *LexisNexis*, and *Thomson Reuters* charge customers high fees for accessing their data using a subscription-based model. More recently, marketplaces like the Azure DataMarketplace[1] has enabled different publishers to sell their data with different subscriptions based on the number of transactions per month. However, none of these companies provide their data in a way such that they can be queried in a federated fashion. They do not provide the means to join multiple data sets, thereby forgoing the complementarities the WoD would enable.

[1] https://datamarket.azure.com/home

This is a serious drawback, because customers are often interested in a specific *combination* of data from different providers that are joined in a certain way.

1.1 Market Incentives for Supplying Data

To create new incentives for providing data in a semantic format, we propose to use a market for querying the WoD. Specifically, we suggest using a double auction to elicit buyers' values for getting an answer to their queries as well as sellers' costs for supplying their data. Our goal in designing the auction is to elicit the (true) values and costs of the market participants such that we can make (approximately) efficient allocations, while satisfying the participants' participation constraints and keeping the market *budget-balanced*. Interestingly, in our domain, satisfying these design goals is highly non-trivial.

One argument against the marketplace's viability is that caching/duplication of data and Bertrand competition will drive prices down to zero, making it uninteresting for sellers to even enter the market. In practice, however, many data providers can sell information due to 1) frequent content changes (e.g., airplane schedules, today's movies, stock prices), 2) ever-increasing content (e.g., court cases, patent information or pharma information), or 3) licensing restrictions (e.g., restricting re-sale and sometimes even re-use of movies, music, or financial exchange information). Thus, there are many data sources for which our proposed market would be viable.

1.2 Market Design Challenges

The first complicating feature of our domain is that the sellers are selling their *data* instead of just *computational resources*. This means that a buyer who submits a query to the market does not know a priori what he will get in return; i.e., he is effectively "buying something of uncertain value." Concretely, the *goods* for sale in the auction are not fully specified upfront. This requires the buyer to specify his value function, which is a function that defines his value for different outcomes of the auction, for all possible result sets that the market could return.

A second, and even more central challenge in our domain, is the fact that the sellers can only provide *statistics* about the data they are selling (i.e., the goods in the auction); but based on these statistics, the market mechanism can only compute rough *estimates* regarding what result a specific allocation will produce. In particular, the exact value of an allocation will only be known after the sellers have produced the result. In consequence, the *allocation rule* and the *pricing rule* of the market mechanism must operate based on *value and cost estimates* – a distinguishing feature of our domain. We will demonstrate that this feature implies that for a two-sided market mechanism which is otherwise guaranteed to satisfy budget-balance and participation (Parkes, Kalagnanam and Eso, 2001), the participation constraint may now become infeasible to be satisfied, and the budget-balance constraint may simply fail. Thus, one of our main research questions is how we can design a market mechanism with good properties in the presence of these domain-specific challenges.

Based on the query, the buyer's value function, the sellers' cost functions, and the statistics, the market computes the estimated value for the different result sets. The buyer does not specify a value for the different result sets itself as the estimation of this value is a computationally challenging task.

1.3 Overview of Contributions

In this paper, we make the following contributions:

1. We propose the "Query Market," a double auction mechanism for querying the WoD.
2. We show that the *Threshold* rule does not work in our domain due to the uncertainty about the sellers' data.

3. We introduce three *payment correction* rules to design payment rules that always satisfy the *participation* constraint despite the uncertainty in our domain.
4. We evaluate the efficiency of the three payment rules via a computational Bayes-Nash equilibrium analysis (Lubin, Bünz and Seuken (2015)) for multiple market scenarios.

1.4 Related Work

1.4.1 Market-based Approaches towards Resource Allocation in Computational Systems

The idea to use markets to allocate computational resources is almost as old as computers themselves. Already in the 1960s, researchers at Harvard University used an auction-like method to determine who gets access to the PDP-1, the world's first interactive, commercial computer (Sutherland (1968)). Since then, many market-based approaches for computational systems have been proposed.

Early research on market-based scheduling focused on the efficiency of computational resource allocation. Malone, Fikes and Howard (1983) present Enterprise, and show how to achieve an efficient allocation of tasks between multiple LAN-connected nodes, where task processors broadcast requests for bids and bid on tasks. Bids reflecting task completion times. Likewise, Spawn by Waldspurger et al. (1992) utilizes a market mechanism to optimize the use of idle resources in a network of workstations. Van Alstyne, Brynjolfsson and Madnick (1995) investigated the impact of soft factors such as ownership for incentive-provisioning in database systems.

More recently, Lai et al. (2005) proposed Tycoon, a distributed computation cluster, featuring a proportional-share market resource allocation model. The authors claim that an economic mechanism is vital for large scale resource allocation a common problem on the Web. Furthermore, market-based optimizations have proved to be as good or better than traditional allocation methods in grid-computing schedulers. Similarly, Auyoung et al. (2006) demonstrate how profit-aware algorithms outperform non-profit aware schedulers across a broad range of scenarios.

Labrinidis, Qu and Xu (2007) applied market-based optimizations to real-time query answering systems. Stonebraker et al. (1996) proposed a WAN-scale Relational Database Management System with a market-based optimizer instead of a traditional cost-based one. Dash, Kantere and Ailamaki (2009) proposed a market-based approach for cloud cache optimization taking into account a user's value for getting an answer to a query. However, their approach focuses on the cost-side of cloud computing. Koutris et al. (2013) proposed a Market for SQL queries which sells data instead of computational resources for answering queries. They use an arbitrage-free pricing scheme instead of a double auction to calculate payments and do not consider competition between sellers in their analysis. Furthermore, buyers in their market do not specify a value for different results nor do sellers specify estimated statistics about their data, hence, they do not face the problem of value and cost estimation, one of the main issues in our market.

Only recently, a new research field called electronic market design has emerged (Anandalingam, Day and Raghavan (2005)). This field provides computer scientists with the necessary tools from auction theory, mechanism design, and market design, to analyze and design markets for computational resources with the same precision as economists have done with great success, for example, in the multi-billion dollar spectrum auction domain (Cramton (2013)). Our goal in this project is to develop a market-based approach for the WoD that is equally grounded in the mathematical foundations of mechanism design and market design.

1.4.2 Querying the Web of Data

Research on distributed query processing has a long history in the database field. Its traditional concepts were used to provide integrated access to RDF sources distributed on the WoD (Harth et al. (2007), Quilitz and Leser (2008), Erling and Mikhailov (2009)). The drawback of these

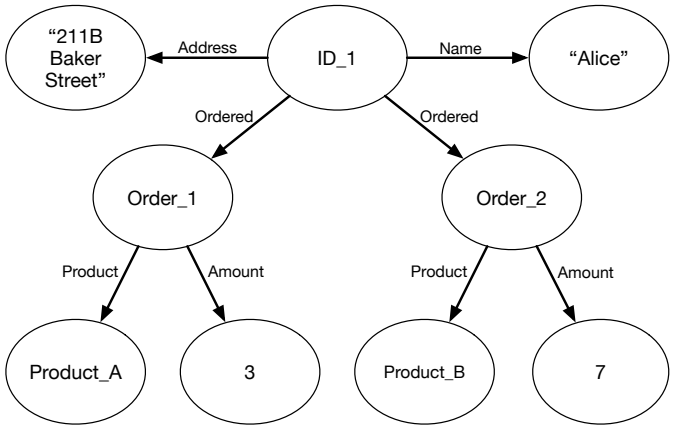

Figure 1: An example dataset, represented as an RDF graph, about a customer "Alice" who ordered 3 items of product A and 7 items of product B.

solutions is that they assume total control over the data distributions, an unrealistic assumption in the open Web. Addressing this drawback, some proposed systems that do not assume fine-grained control but perfect knowledge over the `rdf:type` predicate distribution (Langegger, Wößand Blöchl (2008)) while others proposed to extend SPARQL with explicit instructions controlling where to execute sub-queries (Zemánek, Schenk and Svatek (2007)). Unfortunately, this assumes an ex-ante knowledge of the data distribution on part of the query writer.

More recently, SPLENDID (Görlitz and Staab (2011)) relies on service descriptions and VoID statistics for each endpoint, to perform source selection and query optimisation. In contrast, FedX (Schwarte et al. (2011)) uses no knowledge about mappings or statistics about concepts/predicates but consults all endpoints to determine if a predicate can be answered (caching this information for the future). Finally, Hartig, Bizer and Freytag (2009) describe an approach for executing SPARQL queries over Linked Open Data (LoD) based on graph search. LoD rules, however, require them to place the data on the URI-referenced servers a limiting assumption for example when caching/copying data. Other flexible techniques have been proposed, such as the evolutionary query answering system *eRDF* (Oren, Gueret and Schlobach (2008)), where genetic algorithms are used to learn how to best execute a SPARQL query. However, none of these approaches investigates the economic viability of their proposed solutions.

2 Web of Data (WoD)

One of the main concepts of the WoD is the Resource Description Framework (RDF) (Cyganiak, Wood and Lanthaler (2014)) which models data as *statements* about *resources* (*subject* and *object*) which are linked via a *predicate* that defines the relation between the two resources. A resource denotes something in the universe of discourse (e.g. a physical thing, a website, or an image). Figure 1 shows how data in RDF can be modelled as a graph. Each arrow in the figure represents a predicate pointing from a subject to an object. For example the resource *ID_1* in Figure 1, representing the ID of a customer, is the subject of both, the *Address* and *Name* predicates. The objects of these relations are the literals "221B Baker Street" and "Alice", respectively.

The SQL-like query language SPARQL (Harris and Seaborne (2013)) was proposed as a language to query the Web of Data. Listing 1 shows an example of how such a query can look like. The query asks for name and address of customers, the products ordered by them and the amount. If this query is executed against the dataset in Figure 1 the result from Table 1 will be returned. Each line ending with a "." in the WHERE-clause indicates a triple pattern. The

leading "?" in the query indicates variables. During query processing the query engine searches for binding for these variables such that the result matches the corresponding RDF graph. Each valid match for the variables will produce one row in the result. In a given row, all variables with the same name must be matched to the same resource. For example if the variable "?order" is matched with the resource *Order_1* in the second line of the query in Listing 1 then the variable "?order" in the third line must also be matched with the resource *Order_1*. Hence, in this case the only valid binding for the variable "?amount" is 3 which gives us the first row in Table 1.

```
SELECT ?name ?address ?product ?amount WHERE {
        ?order Product ?product .
        ?order Amount ?amount .
        ?id Ordered ?order .
        ?id Name ?name .
        ?id Address ?address . }
```

Listing 1: A query which asks for name, address, product and amount.

?name	?address	?product	?amount
"Alice"	"221B Baker Street"	Product_A	3
"Alice"	"221B Baker Street"	Product_B	7

Table 1: Result of the query in listing 1.

2.1 Distributed SPARQL processing

If the required data for a query is not located in a single place but distributed over different datasets queries must be processed in a distributed fashion. One of the problems in distributed SPARQL processing is the estimation of the size of a join of triples from different datasets. Figure 2 shows how data could be partitioned over two datasets, A and B. If the same query from Listing 1 is performed over A and B the result will be again as in Table 1. The size of the join is in this case 2 even though there are 4 orders in dataset A and 3 customer names in dataset B. The problem of join estimation is that given some statistics from both individual datasets (in this case the count of triples) it is hard to guess the size of the join of these two datasets. It could be that every one of the four orders matches a customer in B which has 3 different addresses. In this case the result size would be 12, 3 times the same name with 3 different addresses for every order. But it could also be that the data from A and B don't match and hence the result size would be 0. In the field of distributed query processing there are several different approaches to estimate the size of the join without executing the query itself. Each approach has a different trade-off between accuracy and cost (consumption of computational resources).

3 The Query Market

The market mechanism we propose is a double auction that allows a buyer to submit a query and get an answer to his query using datasets provided by different sellers.[2]

[2]To simplify the language, we will use "he" for buyers and "she" for sellers.

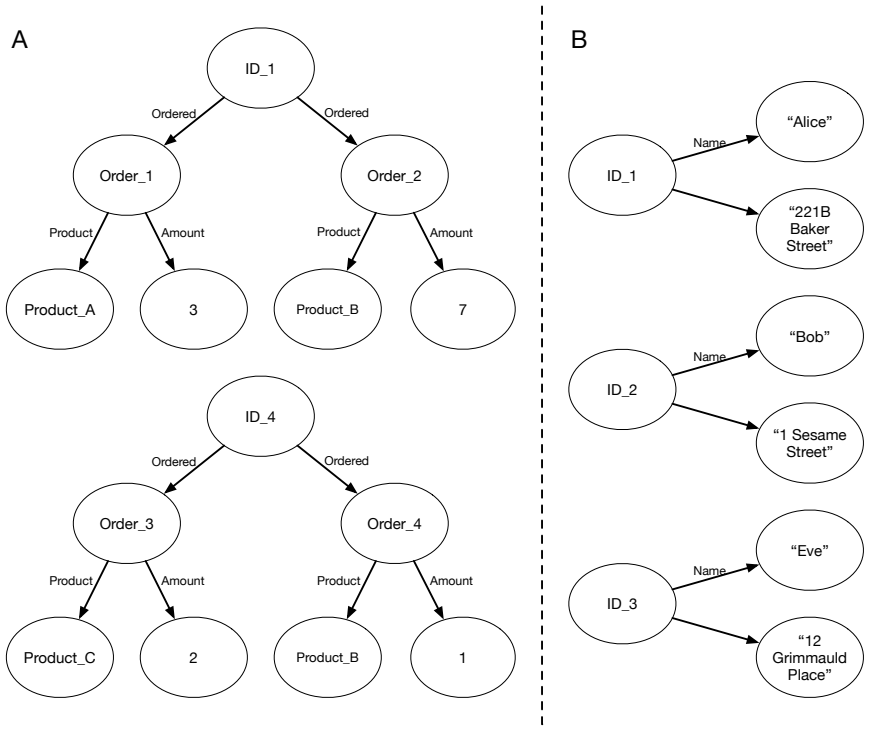

Figure 2: RDF graphs distributed over two datasets, A and B

3.1 Modeling Assumptions

We assume that sellers are only constrained by *what kind of data they can provide*, but that they are not resource-bounded, i.e. that they can potentially answer all the queries for which they have data to provide. This assumption is motivated by the fact that computational resources can nowadays be bought dynamically when needed, for example from cloud services like Amazon EC2. Hence, the cost of the sellers is the marginal cost that occurs answering an additional query which is the cost of the additional resources that need invested to answer the query.

As this implies that buyers are not competing for the same resources, we can hold an independent auction for each buyer's query. Hence, the remaining of the paper will only consider one buyer per auction. In practice multiple auctions run in parallel, one for each query.

Furthermore, we assume that there is no incentive for the sellers to strategize on the statistics they provide for their data. This assumption can be motivated in two ways: (1) the market operator may have the right to audit the statistics of sellers at any point in time, such that incorrect statistics can be detected and penalized; (2) the market operator may directly run a process on the sellers' machines to fetch the statistics in regular time intervals. However, even though the sellers are non-strategic about reporting the statistics, they are only rough (and often incorrect) estimates of what the sellers can deliver, because "precise" estimates might be to expensive to produce. Hence, the statistics delivered by the sellers might be flawed. Additionally, the result of a join between data from different sellers is again an estimate based on their statistics, and thus, imprecise estimates may compound when computing join estimates.

Finally, we assume that the buyer is only strategic about reporting his value function, but submits his true query to the market. The goal of the buyer is to maximize his utility given his query. This means maximizing his value minus the cost he needs to pay for the answer.

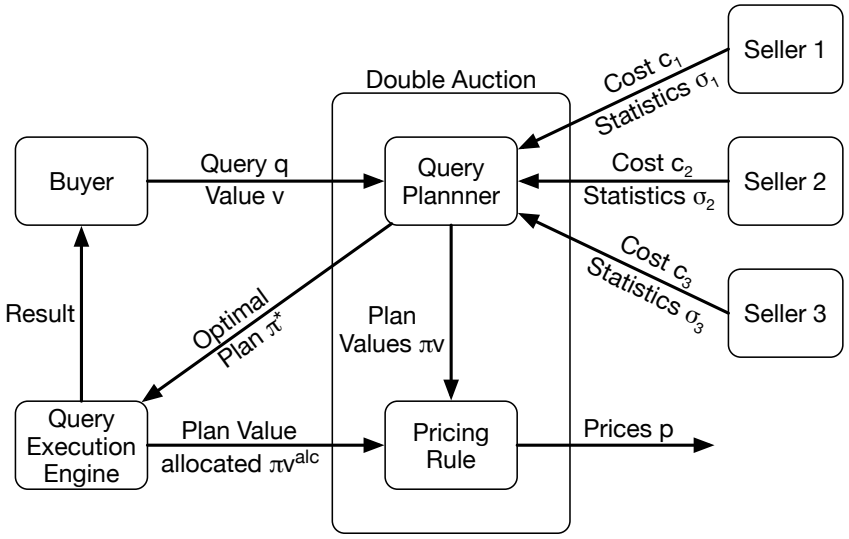

Figure 3: Schematic view of the *Query Market*.

3.2 Formal Model

Figure 3 shows a schematic of our market with three different sellers. We let b denote the buyer and $S = \{s_1, \ldots, s_n\}$ denote the set of sellers in the market. The buyer must submit the query q and the value function v. The pair (q, v) of query and value function is the bid of the buyer. Each seller i needs to provide some statistics σ_i about the data she can provide for the query as well as her cost function c_i. The pair (σ_i, c_i) of statistics and cost function is the bid of seller i.

Example 1. *Consider the query from Listing 1. An example of a value function v would be the number of rows in the result times some constant value, e.g. \$5. In this case the value function would look like:*

$$v = valuePerRow(r) := r \times \$5$$

where r is the number of rows in the result.

For an allocation that produces the result from Table 1 the value would be \$10.

Analogously, a seller can for example specify her cost with respect to the number of triples provided, e.g. \$2. In this case the cost-function c_i would look like:

$$v = costPerTriple(t_i) := t_i \times \$2,$$

n where t_i is the number of triples provided by seller i.

If seller i would provide for a certain allocation 10 triples, her cost would be \$20.

A plan π describes how the query should be answered by the *Query Execution Engine*. This includes the set of sellers that participate in answering the query as well as the concrete instructions for the *Query Execution Engine* how to produce the result for the query.

The *Query Planer* constructs all possible, valid plans that can answer the query q. For each plan π the *Query Planer* calculates the *return set summary* r_i^{est} which is an estimate of what seller i will return if plan π is executed (e.g. the cardinality of the data the seller returns) and combines all the required information to calculate the cost for seller i. Formally, r_i^{est} is the result of r_i which denotes the estimation process and is a function of the plan π and the statistics $\sigma_1, \ldots, \sigma_n$ provided by the sellers:

$$r_i^{est} := r_i(\pi, \sigma_1, \ldots, \sigma_n).$$

The *Query Planer* also calculates the *result set summary* x^{est} which is an estimate of what the buyer will get if plan π is executed (e.g. the cardinality of result of the query) and combines all the required information to calculate the value for the buyer. Note that x^{est} does not denote the actual result of the query as it is not needed for the pricing rule. Formally, x^{est} is the result of x which denotes the estimation process and is a function of the plan π and the statistics $\sigma_1, \ldots, \sigma_n$ provided by the sellers:

$$x^{est} := x(\pi, \sigma_1, \ldots, \sigma_n).$$

Using x^{est} and r_i^{est}, the estimated value $v^{est} := v(x^{est})$ for the buyer and the estimated cost $c_i^{est} := c_i(r_i^{est})$ for seller i can be calculated. These estimates, together with the actual plan π form the vector of *plan values* πv:

$$\pi v := (\pi, v^{est}, c_1^{est}, \ldots, c_n^{est}).$$

Example 2. *Consider a plan π that consists of a join between two sellers, s_1 and s_2. Based on the statistics σ_1 and σ_2, the market estimates x^{est}, r_1^{est}, and r_2^{est}. Given the value and cost functions from example 1, assume that for the join in π seller s_1 needs to provide 10 triples, seller s_2 5 triples, and the cardinality of the result is 8 rows. In this case we have:*

$$r_1^{est} := r_1(\pi, \sigma_1, \sigma_2) = 10,$$

$$r_2^{est} := r_2(\pi, \sigma_1, \sigma_2) = 5,$$

$$x^{est} := x(\pi, \sigma_1, \sigma_2) = 8.$$

With these estimates, the vector of plan values πv can be formed:

$$\pi v := (\pi, v^{est} = \$40, c_1^{est} = \$20, c_2^{est} = \$10).$$

The set of all plan values will be sent to the *Pricing Rule*, and it contains all the information that is required by the *Pricing Rule* to calculate the prices.

At the same time, the *Query Planer* determines the plan π^* that maximizes the estimated social welfare

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\,(SW(\pi, \sigma_1, \ldots, \sigma_n)) = \underset{\pi}{\operatorname{argmax}}\Big(\underbrace{v(x(\pi, \sigma_1, \ldots, \sigma_n))}_{v^{est}} - \sum_i \underbrace{c_i(r_i(\pi, \sigma_1, \ldots, \sigma_n))}_{c_i^{est}}\Big)$$

and sends the allocated plan to the *Query Execution Engine*[3].

The execution of the query produces the result which is sent to the buyer. Using the result of the query execution both, the estimate of the *result set summary* x^{est} and the estimate of the *return set summaries* r_i^{est} of the allocated plan, can be replaced by the actual summaries x^{alc} and r_i^{alc}, where "alc" stands for "allocated". The actual value $v^{alc} := v(x^{alc})$ and costs $c_i^{alc} := c_i(r_i^{alc})$ for the result can now be computed and the vector of the allocated *plan values* $\pi v^{alc} := (\pi^*, v^{alc}, c_1^{alc}, \ldots, c_n^{alc})$ is sent to the *Pricing Rule*.

Example 3. *Assume that the plan from example 2 is the allocated plan. During query execution it might turn out that seller s_1 needs to provide 10 triples for the join but seller s_2 only 4. At the same time, the result set consists only of 5 rows instead of 8. Hence, the vector of the allocated plan values is:*

$$\pi v^{alc} := (\pi^*, v^{alc} = \$25, c_1^{alc} = \$20, c_2^{alc} = \$8).$$

The *Pricing Rule* (further described in Section 4.2) computes the price for the buyer, p_b, and the prices for the sellers, $p_1, \ldots, p_n$. We assume that both, the buyer and the sellers, have a quasi-linear utility function. For the buyer, the utility function is given as follows:

$$u_b(x) = v(x) - p_b \qquad \text{for } x \in \{x^{est}, x^{alc}\}$$

[3]The execution is done in a distributed fashion against the set of sellers from the winning plan using the Federated Query Extension of SPARQL 1.1. See http://www.w3.org/TR/sparql11-federated-query/

and for the sellers as follows:

$$u_i(r) = c(r) - p_i \qquad \text{for } r \in \{r^{est}, r^{alc}\}$$

Remark 1. *Even though the buyer himself does not explicitly specify different bids for different plans, the auction implicitly translates his value function v into a specific bid $v(\pi, \sigma_1, \ldots, \sigma_n)$ which is the value for a given plan π and statistics $\sigma_1, \ldots, \sigma_n$. Since each plan generally consists of multiple sellers, the buyer's value function eventually translates into combinatorial bids for different bundles of sellers. Thus, our auction is a combinatorial double auction.*

4 The Double Auction

When we are designing a double auction, we strive for the following four properties:

1. Efficiency (EFF)
2. Budget-balance (BB)
3. Participation (P)
4. Strategyproofness (SP)

It is well-known, that in an exchange domain, we cannot satisfy all four properties simultaneously. In our domain, we relax strategyproofness and aim for a double-auction that maximizes efficiency, subject to satisfying budget-balance and participation.

4.1 Allocation Rule

The allocation is part of the Query Planner (as described in Section 3) which chooses a plan π^* that maximizes the estimated social welfare $SW = v^{est} - \sum_i c_i^{est}$.

4.2 Pricing Rule

For this section, we generalize and unify the notation for buyers and sellers and denote the set of all *agents* as $I = S \cup \{b\}$. We use index 0 for the buyer and $1...|S|$ for sellers. We let a^* denote an efficient allocation (which corresponds to the plan π^*) and a^*_{-i} an efficient allocation when agent i is excluded. Now, let $w_j(a)$ denote an agent's value for an allocation a, $j \in 0, 1, ..., |S|$, where $w_0(a) = v(a)$ for the buyer, and $w_j(a) = -c_j(a)$ for a seller $j = \{1, ..., |S|\}$. The social welfare for the efficient allocation including all agents is denoted $V^* := \sum_j w_j(a^*)$, and the social welfare for the efficient allocation excluding agent i is denoted $V^*_{-i} := \sum_{j \neq i} w_j(a^*_{-i})$. VCG payments can be computed as follows:

$$p_i^{vcg} := w_i(a^*) - (V^* - V^*_{-i}).$$

We let $\Delta_i^{vcg} := V^* - V^*_{-i}$ denote the *VCG discount* of agent i. Furthermore, we let $\Delta^{vcg} := \{\Delta_0^{vcg}, ..., \Delta_{|S|}^{vcg}\}$. Now, we can equivalently write VCG payments as follows:

$$p_i^{vcg} := w_i(a^*) - \Delta_i^{vcg}.$$

At first sight, VCG payments may seem attractive, because they make truthful reporting of values, a dominant strategy for all agents. However, it is well known that in two-sided market setting like ours, VCG generally leads to a budget-deficit, i.e., the payments made to the sellers can be larger than the payments collected from the buyers (Milgrom, 2007). Formally, the budget-balance constraint is violated.

To address this problem, Parkes, Kalagnanam and Eso (2001) have proposed a family of pricing rules that only *approximate* VCG payments, while ensuring budget-balance. Their idea is to compute payments as $p_i := w_i(a) - \Delta_i$, where Δ_i is a discount assigned to an agent for allocation

a and $w_i(a)$ is a reported value of the agent. Discounts $\Delta = \{\Delta_0, ..., \Delta_{|S|}\}$ are computed to minimize the distance to VCG discounts while satisfying the budget-balance and participation constraints. The *Threshold* pricing rule (Parkes, Kalagnanam and Eso, 2001) can be formulated as the following optimization problem, where the target point is $\Delta^{trg} = \Delta^{vcg}$:

$$\begin{aligned} &\min_{\Delta} L_2(\Delta, \Delta^{trg}) \\ &s.t. \sum_{i \in \{b\} \cup S} \Delta_i = V^* \qquad (budget - balance) \\ &0 \leqslant \Delta_i \leqslant \Delta_i^{trg} \ \ \forall i \in I \qquad (participation) \end{aligned} \tag{1}$$

4.3 Failure of the Participation Constraint

The *Threshold* pricing rule provides budget-balance in a two-sided market domain with no uncertainty about the goods to be sold. However, in our domain, because of inefficiencies which can occur due to uncertainty in statistics provided by the sellers about their data, all of the pricing rules proposed by Parkes, Kalagnanam and Eso (2001) may lead to an infeasible participation constraint in the payment computation problem.

We now provide an example which illustrates how imprecise estimates provided by the sellers can cause problems with the participation constraint. In particular, the following example shows that even VCG can end up violating the participation constraint in a domain with uncertainty.

Example 4. *Let b be the buyer and $S = \{s_1, s_2, s_3\}$ three different sellers. Assume two possible plan values are generated by the Query Planer for b's query. The first plan value $\pi v_1 = \{\pi_1, v^{est}(\pi_1) = 3, c_1^{est}(\pi_1) = 0.5, c_2^{est}(\pi_1) = 2\}$, the second one $\pi v_2 = \{\pi_2, v^{est}(\pi_2) = 6, c_2^{est}(\pi_2) = 1, c_3^{est}(\pi_2) = 4\}$. If there was no uncertainty in the domain, an efficient allocation would choose πv_2 resulting in VCG payments $p_b^{vcg} = 5$, $p_{s_1}^{vcg} = 0$, $p_{s_2}^{vcg} = -2$, $p_{s_3}^{vcg} = -4.5$. While not being budget-balanced, these payments provide participation.*

Now assume that there is an uncertainty and after πv_2 is allocated we have: $\pi v_1 = \{\pi_1, v^{est}(\pi_1) = 3, c_1^{est}(\pi_1) = 0.5, c_2^{est}(\pi_1) = 2\}$, $\pi v_2^{alc} = \{\pi_2, v^{alc} = 3, c_2^{alc} = 1, c_3^{alc} = 2\}$. VCG payments computed for this actual allocation are $p_b^{vcg} = 3$, $p_{s_1}^{vcg} = 0$, $p_{s_2}^{vcg} = -1$, $p_{s_3}^{vcg} = -1.5$. It's clear now that the participation constraint for s_3 is not satisfied anymore because $|p_{s_3}^{vcg}| < c_3^{alc}$.

If VCG violates the participation constraint, then payment rule (1) becomes infeasible in our domain. Thus, we need to design payment correction rules to tackle this problem.

4.4 Payment Correction Rules

We now propose three different payment correction rules and study which one leads to higher efficiency (in equilibrium). As mentioned before, because of uncertainty, our mechanism might sometimes make an inefficient allocation (even given truthful value reports). In this case, one cannot rely on VCG because of negative VCG discounts which lead to infeasibilities in the participation constraints (as explained in Section 4.3). A similar problem was faced by Goetzendorf et al. (2015), where the authors just trimmed such infeasibilities. We could try to apply this idea to our domain and use the trimmed VCG discounts as a target point Δ^{trg} in (1). While this new target guarantees satisfaction of the participation constraint, it is not clear whether it provides good incentives (and thus good efficiency). For this reason, we also define two alternative methods for constructing a target point Δ^{trg}. For this we let π^* denote the plan for the allocation chosen by the Query Planer. We the define $V_{est}^* := v(x(\pi^*, \sigma_1, \ldots, \sigma_n)) - \sum_i c_i(r(\pi^*, \sigma_1, \ldots, \sigma_n))$ as the total value of all agents in this allocation (before query execution), and $V_{est,-i}^* := v(x(\pi_{-i}^*, \sigma_1, \ldots, \sigma_n)) - \sum_i c_i(r(\pi_{-i}^*, \sigma_1, \ldots, \sigma_n))$ as the total value of all agents in the allocation where agent i is excluded. Our three payment correction rules are:

1. `PC-TRIM`: Compute VCG discounts using x^{alc} and r^{alc} for the allocated plan, given a possibly non-efficient allocation (similarly to Goetzendorf et al. (2015)). Then trim infeasibilities in the participation constraints by increasing all negative Δ_i to be zero and use the resulting discounts as a new target point Δ^{trg}.
2. `PC-PENALTY`: We let $V^*_{alc,i}$ denote the social welfare for the allocated plan π^*, where we use actual summaries (based on the result from the executed query) for seller i, but use the reported statistics for all other sellers. Then we compute VCG discounts based only on the statistic reported by sellers. Next we reduce those discounts which correspond to sellers who have provided wrong estimates by a penalty factor $\alpha_i = \frac{V^*_{est} - V^*_{alc,i}}{V^*_{est}}$, for all i involved in π^*. The penalty factor reflects the harm caused by agent i to the efficiency by providing her imprecise estimates.
3. `PC-VCG`: Compute VCG discounts using the (possibly wrong) estimates x^{est} and r^{est} from the Query Planner, instead of x^{alc} and r^{alc}, even though the query has already been executed, and use those estimate-based VCG discounts as a target point Δ^{trg}.

The resulting payment rules for our market are derived from the *Threshold* payment rule by applying the aforementioned payment correction procedures. However, it is not strictly necessary to use *Threshold* as an underlying payment computation rule. Instead many other VCG approximation techniques can be utilized in combination with the proposed payment correction rules, for example, *Small*, *Large*, *Fractional* (Parkes, Kalagnanam and Eso (2001)) etc.

To check which payment rule provides better efficiency, we compare all three of them in a double auction setup when all agents play according to the Bayes-Nash equilibrium (BNE) of the mechanism.

5 Quantitative Evaluation

To study the efficiency of the three proposed payment rules, we need to make an assumption regarding the agents' behavior in the market. Assuming agents to be rational, we allow them to strategize and act as if they would try to maximize their expected utilities. This allows us to naturally restrict our analysis to only setups when all agents play according to their *equilibrium strategies*. In other words, we are interested in finding a strategy profile for all agents which would maximize the expected utility of every agent assuming that all other agents also play according to the strategies in this strategy profile. This approach allows us to capture the strategic behavior of the agents (due to non-strategyproof payment rules) and evaluate efficiency loss caused by the agents' strategic behavior. Given that a full information assumption seems highly unrealistic in this domain, we adopt a *Bayes-Nash equilibrium* analysis approach.

However, theoretical evaluation of the BNE can be an extremely difficult task. Goeree and Lien (2014) derived a BNE for a simple combinatorial auction setup with only three agents and two items. Ausubel and Baranov (2010) derived a BNE for a similar setup but taking into account correlated values of agents. However, there are no any further results for neither more complicated combinatorial auctions nor for double auctions. In addition to that, our market usually involves much more than just 3 agents which makes the theoretical analysis even harder. Thus, instead of deriving the equilibrium analytically we try to approximate it computationally. Section 5.1 describes more precisely how a BNE can be approximated. Section 5.2 introduces the simulated WoD domain and describes the quantitative evaluation framework in detail. In conclusion Section 5.3 provides a detailed explanation of the results of our evaluations.

5.1 Approximate BNE Computation

To perform the quantitative evaluation of our market we compute (ϵ, δ)-BNEs using a method described by Lubin, Bünz and Seuken (2015). This method is based on an iterative best-response search procedure.

The general approach for BNE approximation implements an idea of a *Fictitious Play* and can be described as a two stage procedure. First, agents are grouped into one of n_b bins. We use only $n_b = 2$ bins: one for buyers and one for sellers. Agents within same bin are assumed to play the same strategy. We also assume that the strategy space for a buyer with value v can be represented as $v \cdot s, s \in [0, 1]$ implying that the buyer can only underbid but not overbid. Similarly, the strategy space for a seller with a cost c is $c \cdot \frac{1}{s}, s \in [0, 1]$ which means that the seller only overreport his costs.

The second stage is an iterative process during which for every bin a best response strategy is computed, i.e., a strategy which is when being played by every agent within the bin maximizes the total utility of all agents in the bin (assuming the strategies of other agents are fixed). At this stage on every iteration `nSamples` games are sampled using known distributions of types of all agents. Then, each agent within a particular bin plays different strategies while strategies of all other agents are fixed. Every such a game contributes into the overall utility of the bin which is computed as a sum of utilities of all agents from the bin. This reduces the aforementioned problem of the best response strategy search to a global stochastic optimization problem where the objective is to maximize the total utility for every bin while varying strategies of agents within the bin. Clearly, the number of such global optimization problems per iteration is equal to the number of bins. When optimal strategies for bins are computed, we apply them for every agent in corresponding bins by allowing them to play a convex combination of their current strategy and the newly found best response. This leads to an update of the strategy, or *shaving factor*, for every bin and followed by the next iteration.

The process terminates when an (ϵ, δ)-BNE is identified, i.e., when playing the best response rather than the (ϵ, δ)-BNE strategy does not improve bins' overall utilities more than by a factor of $1 + \epsilon$. More formally,

Definition 1. *A strategy profile $s^* = (s_1^*, ..., s_{nBins}^*)$ is an (ϵ, δ)-Bayes-Nash equilibrium if for every bin $i \in \{1, ..., nBin\}$*

$$\frac{\mathbb{E}_{-i}[u_i(br_i, s^*_{-i})]}{\mathbb{E}_{-i}[u_i(s_i^*, s^*_{-i})]} \leqslant 1 + \epsilon,$$

and

$$||s_i^* - br_i(s^*_{-i})||_2 \leqslant \delta,$$

where br_i denotes a best response of the i-th bin and u_i is a total utility of the i-th bin.

We use $\epsilon = 1\%$, $\epsilon = 2.5\%$ and $\epsilon = 3\%$ for the scenario with three, five and ten sellers respectively. For all scenarios we assume $\delta = 0.1$. We also varied `nSamples` from 1000000 for a small setup with only three and five sellers to 500000 for the setup with 10 sellers. To solve the described global optimization problem of a best response search we utilize multistart method with 10 starting points generated randomly from $U[0, 1]$ combined with a pattern search local optimization procedure.

5.2 Domain Description

To evaluate the market we generate three different WoD domains and thus simulate three different market scenarios. The first scenario includes a single buyer b_1 and three sellers s_A, s_B and s_C. For this scenario we assume that the Query Planner always generates two plans π_1 and π_2. Each plan involves two different sellers chosen randomly from a uniform distribution. We denote $t_X^{est}(\pi_i)$ the estimated (and possibly imprecise) number of triples which a seller s_X reports she can provide for the plan π_i, $X \in \{A, B, C\}$, $i \in \{1, 2\}$. The numbers of triples provided by different sellers is chosen uniformly from $[1, 10]$. Even though such a case with only three sellers is not very likely to happen in a real Query Market where the number of sellers is usually large, it brings out some interesting properties of the market. More specifically, we use this scenario to study what happens with the market if it has very influential agents.

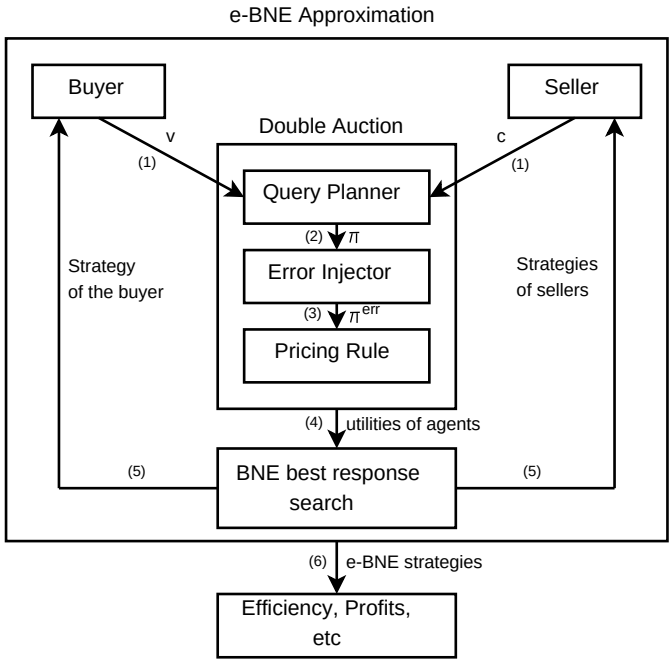

Figure 4: Evaluation framework.

The second scenario includes a single buyer and five sellers. For this scenario we again generate two different plans each involving two different sellers chosen randomly from a uniform distribution. By comparing the resulting strategies of this scenario with the corresponding strategies from the previous one, we study how the factor of the number of sellers alone influences on the efficiency of our market.

The third scenario is more typical for the Query Market and includes a single buyer and 10 sellers. For this scenario we generate five different plans. Similarly to previous two scenarios each plan involves two different sellers chosen randomly from a uniform distribution. A high diversity of sellers and a small number of sellers involved in a single plan reduces the effect of individual sellers.

For all domains, we assume the agents' value and cost functions are linear in the cardinality of data. Thus, we only need a value and cost factors, v and c_i, $i \in S$ respectively to be reported by agents. We assume that v and c_i are drawn independently from a uniform distribution $U[0,1]$, $(i \in S)$.

When a plan is allocated an estimation error can be injected into the plan. To model estimation errors (or uncertainty), we randomly select a seller $X \in \{A, B, C, ...\}$ from a uniform distribution over the set of sellers involved in the allocated plan π^*. We then generate $t_X^{alc} \sim \mathcal{N}(t_X^{est}, \sigma)$ truncated at 0 instead of t_X^{est}. Furthermore, we assume that a seller can only overestimate t_X^{est}, i.e., $t_X^{est} \geqslant t_X^{alc}$; and if $t_X^{alc} > t_X^{est}$, we use $t_X^{alc'} = t_X^{est} - (t_X^{alc} - t_X^{est})$ instead. Underestimation of the number of allocated triples should not be a problem as in this case the seller can simply discard extra triples which are not required by the plan. Additionally, by varying σ we study how the degree of uncertainty in the domain influences on agents' incentives and the resulting market efficiency.

For the three described scenarios, we first approximate the Bayes-Nash equilibrium for every payment rule. Then we feed the mechanism with all agents playing according to the BNE into a benchmarking module to measure the following characteristics:

Type of statistic	Payment rule	1 Buyer, 10 Sellers							
		Strategies, %			% of VCG profits		rate of BB	BB	Partici-
		Buyer	Sellers	Efficiency	Buyer	Sellers	violations,%	deficit	pation
w/o uncertainty	VCG	100	100	1.00	100	100	72	0.87	✓
	THRESHOLD-TRUTH	100	100	1.00	69	76	0	0	✓
	THRESHOLD	76	147	0.39	48	42	0	0	✓
w. uncertainty	VCG-TRUTH	100	100	0.86	89	80	66	0.80	×
	VCG	98	109	0.77	80	77	64	0.89	×
	PC-TRIM	78	142	0.37	43	37	1.8	0.005	✓
	PC-PENALTY	84	135	0.46	46	44	1.6	0.003	✓
	PC-VCG	80	136	0.41	45	41	1.7	0.004	✓

Table 2: Results of a computational BNE analysis for all pricing rules in a uncertainty-free and uncertain setting for the scenario with one Buyer and 10 Sellers.

- The total efficiency of the market relative to the total efficiency of the market if VCG is used to compute payments and there is no uncertainty in the domain.
- Profits of buyers and profits of sellers relative to their profits if VCG is used for payment computation in a domain without uncertainty.
- The rate of budget-balance violations, i.e., how often does it happen that the budget-balance constraint is violated.
- The amount of budget-balance deficit w.r.t. the total efficiency of the market, i.e., the ratio of the budget-balance deficit to the total social welfare.

The general workflow of our evaluation framework is presented in Figure 4. First, buyer's and sellers' types are sampled and their value and cost functions are submitted into the auction (1). Second, a set of plans is generated by the Query Planner (2). Then an allocation happens followed by an injection of an estimation error into the allocated plan (3). Payments are computed on the next step. Resulting utilities of agents are used in a best response search procedure to identify optimal strategies for agents on the current iteration. The process repeats (5) using strategies computed on the previous iteration until ϵ-BNE is found and then these BNE strategies are fed into the efficiency, profits and budget-balance evaluation module (6). The efficiency module in its turn generates 100,000 markets with agents acting according to the identified BNE and computes the total efficiency, profits of agents, number of budget-balance constraint violations and the budget amount required to cover these violations etc.

5.3 Results

Table 2, Table 3 and Table 4 show our evaluation results. We have considered two different cases (with and without uncertainty): within the uncertainty-free setting, we consider `VCG`, `THRESHOLD-TRUTH`, and the standard `THRESHOLD` rule. The `THRESHOLD-TRUTH` rule is simply the standard Threshold rule where instead of playing their BNE strategies, we force all agents to play truthfully. While being very artificial, this mechanism gives us an upper bound on the agents' profits for a budget-balanced mechanism in the uncertainty-free setting.

For the domain with uncertainty, we consider five pricing rules: `VCG-TRUTH` (a benchmark which obtains true values, but uncertain reports from sellers and uses VCG as a payment rule), `VCG` (a benchmark which computes VCG using uncertain reports), and then the three price correction (PC) rules `PC-TRIM`, `PC-VCG` and `PC-PENALTY`. Note that `VCG` for the setup with uncertainty is not truthful anymore due to inefficient allocations and thus the agents will play their BNE strategies.

5.3.1 Market Scenario #1: 1 Buyer and 10 Sellers

Consider Table 2, where we present the results for market scenario #1 with 1 buyer and 10 sellers. This scenario is meant to represent a typical WoD domain with lots of sellers, where no individual seller is pivotal for trade to happen.

The first observation we make is that `THRESHOLD` only achieve 39% of the efficiency of `VCG`. This result is not surprising, because while `VCG` is strategyproof, `THRESHOLD` is not. As we can see, under `THRESHOLD`, the agents start manipulating (the buyer under-reports by 24% and the sellers over-report by 47%), which results in lost trades, and thus explains the lower efficiency of `THRESHOLD`.

Next, we compare `VCG` in the domain without uncertainty to `VCG-TRUTH` in the uncertain setting. Here, the 14% smaller efficiency of `VCG-TRUTH` indicates how much efficiency we lose due to imprecise estimates and not because of agents' manipulations. Then we compare `VCG` with `VCG-TRUTH` in the domain with uncertainty. Here we observe that `VCG` loses an additional 9% points over and above the efficiency loss that `VCG-TRUTH` already incurred, which must now partly be due to the strategy manipulations of the agents. Added together, the uncertainty injection and the agents' strategic manipulations lead to a 23% efficiency loss, when compared to `VCG` in the uncertainty-free setting.

Finally, we consider our three payment correction rules, `PC-TRIM`, `PC-PENALTY` and `PC-VCG`, whose efficiency is between 0.37 and 0.46. Let's first compare the resulting efficiency to that of `VCG-TRUTH`, the efficiency upper-bound for the market with uncertain reports. For example, `PC-VCG` achieves 48% of the efficiency that `VCG-TRUTH` achieves. Now let's consider `PC-TRIM`, which actually uses `VCG` from the setting with uncertainty as its target point in the price computation algorithm. Thus, the efficiency of `PC-TRIM` is upper-bounded by the efficiency of `VCG` in the setting with uncertainty. Here, we obtain that `PC-TRIM` achieves 48% of the efficiency of `VCG` in the setting with uncertainty. Note that it is not surprising that in some cases (like for `PC-TRIM`) our payment correction rules do not achieve the same share of the maximum achievable efficiency that `THRESHOLD` is able to achieve in the uncertainty-free setting, because (1) we need to correct the participation constraint, which may lead to a further efficiency loss, and (2) in the domain with uncertainty, we are already starting with a target point $\Delta^{trg} = \Delta^{vcg}$ which already leads to significant strategy behavior by the agents and an efficiency loss of $(1 - \frac{0.77}{0.86}) \cdot 100\% = 11\%$. This suggests that the choice of Δ^{vcg} as a target point in the uncertain environment might be sub-optimal - in contrast to the uncertainty-free setting, where generally using Δ^{vcg} as the target point provides good incentives and thus leads to high efficiency. This effect can be seen more clearly in the scenario with smaller number of agents (see, for example, section 5.3.3).

We now look at the profits of the agents in the market, which in Table 2 are normalized relative to the profits the agents obtain under VCG in the uncertainty-free setting. We see that the buyer obtains between 43% and 46% of his VCG profits, while the sellers obtain between 37% and 44% of their VCG profits. Importantly, all three payment correction rules lead to almost identical profits. Furthermore, we see that under `PC-TRIM`, the buyer obtains a profit of $\frac{0.43}{0.89} \cdot 100\% = 48\%$ of the maximal attainable profit under `VCG-TRUTH`; and the seller obtains a profit of $\frac{0.37}{0.80} \cdot 100\% = 46\%$ of the maximum attainable profit under `VCG-TRUTH`. Thus, these results are relatively close to the 48% and 42% which `THRESHOLD` achieves in the setting without uncertainty.

5.3.2 Market Scenario #2: 1 Buyer and 5 Sellers

In Table 3 we present the results for market scenario #2 with 1 buyer and 5 sellers. This scenario represents an intermediate case between the most typical for the semantic web scenario #1 and an almost extreme scenario #3 with very influential agents.

In this case we obtained strictly worse strategies for the buyer and sellers comparing with

Type of statistic	Payment rule	1 Buyer, 5 Sellers							
		Strategies, %			% of VCG profits		rate of BB	BB	Partici-
		Buyer	Sellers	Efficiency	Buyer	Sellers	violations,%	deficit	pation
w/o uncertainty	VCG	100	100	1.00	100	100	60	1.51	✓
	THRESHOLD-TRUTH	100	100	1.00	47	73	0	0	✓
	THRESHOLD	67	172	0.16	28	15	0	0	✓
w. uncertainty	VCG-TRUTH	100	100	0.86	86	85	58	1.45	×
	VCG	97	102	0.79	85	78	56	1.48	×
	PC-TRIM	72	166	0.18	26	16	1.2	0.01	✓
	PC-PENALTY	72	161	0.19	28	16	1.3	0.01	✓
	PC-VCG	71	163	0.18	27	16	1.2	0.01	✓

Table 3: Results of a computational BNE analysis for all pricing rules in a uncertainty-free and uncertain setting for the scenario with one Buyer and 5 Sellers.

those of scenario #1. This shows that the higher number of agents and available plans can considerably decrease manipulability of the market and raise its efficiency. In our case the low number of agents results in more than halved efficiency for all proposed payment correction rules. For example, we have $49\% = \frac{0.18}{0.37} \cdot 100\%$ of efficiency drop for PC-TRIM, $41\% = \frac{0.19}{0.46} \cdot 100\%$ for PC-PENALTY, and $44\% = \frac{0.18}{0.41} \cdot 100\%$ for PC-VCG. However, such a dramatic drop in efficiency is not specific only for our payment correction rules: our experiments also provide a similar trend for the THRESHOLD rule in the setup with no uncertainty. In this case we have $42\% = \frac{0.16}{0.38} \cdot 100\%$ smaller efficiency when decreasing the number of agents and available plans.

Perhaps the most surprising result is the fact that the efficiency of all payment correction rules in the setting with uncertainty (around 18%) is higher than the efficiency of THRESHOLD in the uncertainty-free setting (16%). This finding is not easy to explain, because we would typically expect that the uncertainty in the domain leads to a *reduction* in the efficiency, as we have observed it in the previous scenario. Our only explanation is that THRESHOLD, even though it is using VCG discounts as a target point, is only the optimal payment rule in an ex-post sense. However, in a Bayesian setting, only little is known about which payment rule is best (Lubin and Parkes (2009)). Our current conjecture is that our payment correction rules have coincidentally picked out such an attractive target point that, evaluated in BNE, it leads to even higher efficiency than THRESHOLD does in the uncertainty-free setting. However, further analysis is needed to evaluate this conjecture.

5.3.3 Market Scenario #3: 1 Buyer and 3 Sellers

Now consider Table 4, where we present the results for market scenario #3. Remember that this is an extreme scenario where every seller is very influential; ever seller can potentially be pivotal; and where one of the three sellers is required for all of the plans (i.e., he is a "monopolist").

The most important observation from Table 4 is that the efficiency of all three payment correction rules is now only around 15%, and thus much lower than in the scenario with 10 sellers (where it was around 40%). This can of course be explained by the fact that in this scenario, the sellers have become much more pivotal, which has increased their incentives to manipulate. While previously, the sellers where overbidding by approximately 40%, they are now overbidding by at least 63% and up to 75% (for PC-PENALTY). Remember that PC-PENALTY only penalizes sellers, and in particular those sellers that have provided imprecise estimates that have led to an inefficient allocation. Thus, the buyers can usually expect to receive high VCG discounts under this rule, and consequently play almost truthfully. The sellers, in contrast, have very large incentives to manipulate.

Type of statistic	Payment rule	1 Buyer, 3 Sellers							
		Strategies,%			% of VCG profits		rate of BB	BB	Partici-
		Buyer	Sellers	Efficiency	Buyer	Sellers	violations,%	deficit	pation
w/o uncertainty	VCG	100	100	1.00	100	100	59	1.55	✓
	THRESHOLD-TRUTH	100	100	1.00	45	73	0	0	✓
	THRESHOLD	71	178	0.17	26	14	0	0	✓
w. uncertainty	VCG-TRUTH	100	100	0.96	95	96	59	1.52	×
	VCG	98	112	0.80	84	82	59	1.54	×
	PC-TRIM	67	163	0.15	25	12	1.1	0.013	✓
	PC-PENALTY	70	175	0.15	22	11	1.1	0.014	✓
	PC-VCG	69	163	0.16	25	13	1.2	0.012	✓

Table 4: Results of a computational BNE analysis for all pricing rules in a uncertainty-free and uncertain setting for the scenario with one buyer and three sellers.

5.3.4 Budget-balance Violations and Budget Deficits

Remember that our goal was to design a budget-balanced double auction mechanism. As mentioned before, it is well known that VCG is not budget-balanced in a two-sided market. We can now put a number on how severe the budget-balance violations of VCG are in our simulated domains. In Table 2, we see that in the uncertainty-free setting, VCG led to a budget-balance violation in about 72% of the cases. In Table 4, VCG led to a budget-balance violation in about 59% of the cases.

The THRESHOLD rule is explicitly designed to be budget-balanced, but only works in the uncertainty-free setting. For this reason, we need our new price correction rules, i.e., the only rules which are always guaranteed to satisfy the participation constraint in the uncertain setting. However, even those three rules can sometimes still lead to a budget-balance violation when a plan is allocated whose total costs are larger than its value, which may happen due to imprecise estimates. These kinds of budget-balance violations cannot be avoided in our market due to the nature of the WoD. But fortunately, as we can see in Table 2 and 4, such budget-balance violations occur very infrequently for our payment correction rules: between 1.6% and 1.8% in the market with 10 sellers, and between 1.1% and 1.2% in the market with 3 sellers.

Finally, when these budget-balance violations do occur, then the amount of budget deficit that our payment correction rules exhibit is minimal. For example, in Table 2, we see that the average budget deficit for our rules is between 0.003 and 0.005, which is almost negligible compared to a budget deficit of 0.87 of VCG (all of these numbers are relative to total social welfare). In Table 4 the average budget deficit of our rules is between 0.012 and 0.014, compared to a average budget deficit of 1.55 of VCG.

Of course, in practice, we would have to find some way to cover these budget deficits. Given the almost negligible size of the budget deficit, a practical approach is to simply increase the payments charged to buyers by a correspondingly small amount on all other trades where no budget-balance violations occur. Developing a practical method for automatically calibrating these additional "taxes" such that the market operator always has enough budget to cover the expected budget deficit is subject to future work.

5.3.5 Impact of the Degree of Uncertainty on the Market Efficiency

By analyzing Table 2 and Table 3 we see that the efficiency of the proposed payment rules in the domain with uncertainty is usually higher than the efficiency of a *Threshold* rule in a domain without uncertainty. At the same time agents in an uncertain domain manipulate less than in the case when all estimates are precise. This can be caused either by a more attractive target

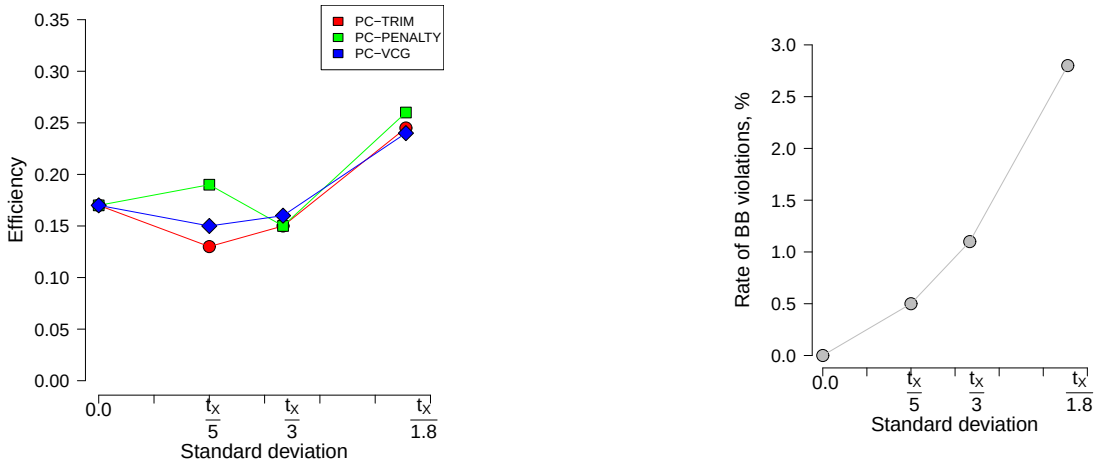

Figure 5: Efficiency and the rate of budget balance violations for different degrees of uncertainty in the scenario with a single buyer and three sellers

point Δ^{trg} in the domain with uncertainty or by the fact that in an uncertain domain it is easy for an agent to lose some of its utility by manipulating because of a more unpredictable market behavior. To check if a higher uncertainty can really increase the efficiency of the market we perform a set of experiments with different levels of uncertainty injected into the market.

To manipulate the degree of uncertainty of our market we vary the σ parameter at the error injection step. Remember, that the error is injected in the following way $t_X^{alc} \sim \mathcal{N}(t_X^{est}, \sigma)$, where t_X^{est} is the estimated number of triples reported by a seller s_X, $X \in \{A, B, C, ...\}$ (see Section 5.2). Thus, by increasing σ we increase the uncertainty in the domain. In all our experiments we parametrize $\sigma = \frac{t_X^{est}}{\alpha}$, where $\alpha \in \mathbb{R}_+$ is a parameter used to manipulate the uncertainty level. We perform a number of experiments with $\alpha = 1.8$, $\alpha = 3$, $\alpha = 5$, and $\alpha \to \infty$ (which corresponds to a domain without uncertainty). Note that in the case with $\alpha \to \infty$ (and, consequently, $\sigma = 0$) all the proposed payment rules are essentially equivalent to the standard *Threshold* rule.

Figures 5-7 illustrate the dependency of the efficiency and the rate of budget balance violations on the uncertainty level of the domain. For all considered scenarios with different number of agents we see a positive trend in efficiency when σ increases. In the case with three sellers there is a small drop in efficiency when increasing σ from 0 to $\frac{t_X}{3}$. We explain this by the fact that in the domain with very influential (or monopolistic) sellers agents can still manipulate a lot even despite of a small uncertainty in the domain. However, when σ is large enough it becomes difficult to manipulate even for influential agents and thus we again have a positive trend in efficiency when $\sigma > \frac{t_X}{3}$.

It is misleading, however, to think that we can always increase the efficiency of the market by injecting more uncertainty in the domain. The problem which arises here is that when we increase the uncertainty in the domain we also increase the number of budget balance violations (see the right-side Figures 5-7). This is not surprising as larger differences between t^{est} and t^{als} (due to larger σ) lead to a higher amounts of inefficient allocations with a total cost of a plan larger than the value of a buyer for the plan. Thus, there is a trade-off between a high efficiency and a budget balance property. Important also that for the large number of sellers the trend of

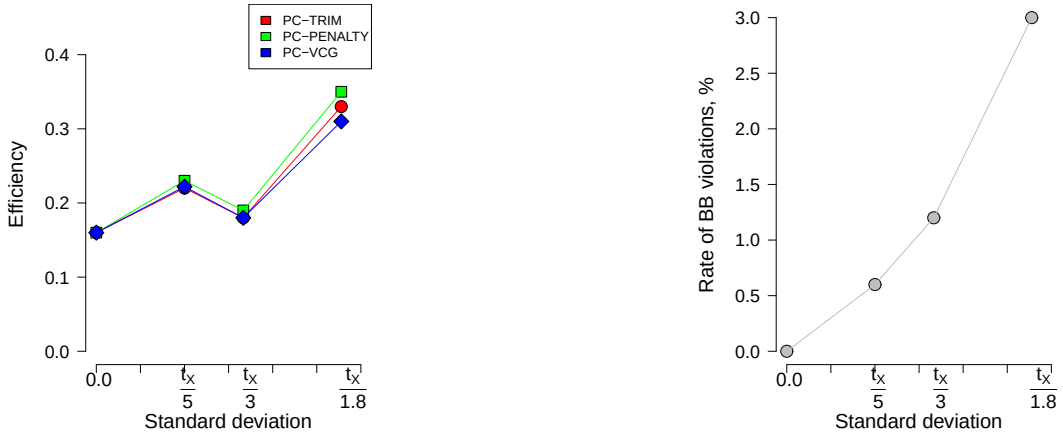

Figure 6: Efficiency and the rate of budget balance violations for different degrees of uncertainty in the scenario with a single buyer and five sellers

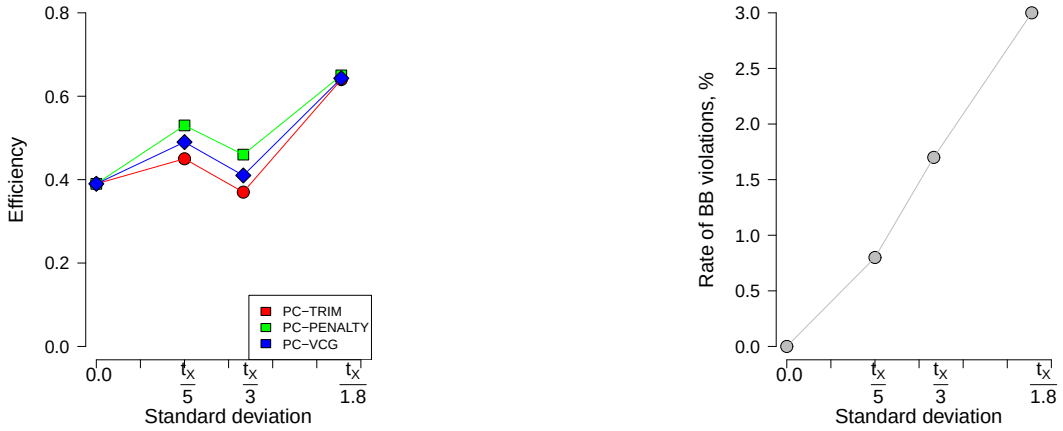

Figure 7: Efficiency and the rate of budget balance violations for different degrees of uncertainty in the scenario with a single buyer and 10 sellers

budget balance violations is almost linear. This means that small perturbations of the uncertainty in statistics provided by sellers does not lead to a dramatic growth of the budget required to cover all inefficiencies occurred due to these imprecise estimates.

6 Conclusion

In this paper, we have proposed a double auction for querying the WoD which handles the fact that the input to both, the allocation and pricing rules, are uncertain due to statistical estimation errors. We have proposed a general market framework for this domain, and introduced a formal model for the double auction. It turns out that a particular challenge in our domain was to design a pricing rule that satisfies the participation constraint despite uncertainty. In particular, the *Treshold* rule, which is guaranteed to satisfy participation in standard double auction environments, can now sometimes lead to an infeasible pricing problem. Towards this end, we have proposed three payment correction rules which always guarantee the participation constraint.

We have evaluated the three rules via a computational Bayes-Nash equilibrium analysis. Interestingly, we have found that incentives, the efficiency, and profits of our new rules are relatively competitive with the *Threshold* rule in a uncertainty-free setting, despite the significant influence of imprecise estimates on the market. We have also evaluated the impact of the number of sellers (and their influence), and found that, consistent with intuition, a larger number of sellers leads to higher efficiency, less manipulation, and higher profits. Furthermore, we have studied the amount of budget required to cover the budget deficit which still occurs, even under our price correction rules, and found this to be almost negligible. Overall, our evaluation results look promising toward implementing an auction-based market for the WoD.

In future work, we are planning to investigate how relaxing some of our assumptions (like those regarding the value and cost functions of the agents) may affect the efficiency, profits and incentive properties of the market. Furthermore, we are going to investigate the impact of a more realistic error injection module, by incorporating a real Query Planner as well as real data stores into our simulation setup.

References

Anandalingam, G., Robert W. Day, and S. Raghavan. 2005. "Landscape of Electronic Market Design." *Management Science*, 51(3): 316–327.

Antoniou, Grigoris, and Frank van Harmelen. 2004. *A Semantic Web Primer*. Cambridge, Massachusetts, London, England: The MIT Press.

Ausubel, Lawrence M., and Oleg V. Baranov. 2010. "Core-Selecting Auctions with Incomplete Information."

Auyoung, Alvin, Laura Grit, Janet Wiener, and John Wilkes. 2006. "Service Contracts and Aggregate Utility Functions." In *In Proceedings of the IEEE Symposium on High Performance Distributed Computing*.

Cramton, Peter. 2013. "Spectrum Auction Design." *Review of Industrial Organization*, 42(2): 030–190.

Cyganiak, Richard, David Wood, and Markus Lanthaler. 2014. "RDF 1.1 Concepts and Abstract Syntax." *http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/*.

Dash, Debabrata, Verena Kantere, and Anastasia Ailamaki. 2009. "An Economic Model for Self-tuned Cloud Caching." In *Proceedings of the 2009 IEEE International Conference on Data Engineering*.

Erling, O, and I Mikhailov. 2009. "RDF Support in the Virtuoso DBMS." OpenLink Software.

Goeree, Jacob, and Yuanchuan Lien. 2014. "On the Impossibility of Core-Selecting Auctions." *Theoretical Economics*. Forthcoming.

Goetzendorf, Andor, Martin Bichler, Pasha Shabalin, and Robert W. Day. 2015. "Compact Bid Languages and Core Pricing in Large Multi-item Auctions." In *Management Science*.

Görlitz, Olaf, and Steffen Staab. 2011. "SPLENDID: SPARQL Endpoint Federation Exploiting VOID Descriptions." In *Proceedings of the 2nd International Workshop on Consuming Linked Data*. Bonn, Germany.

Harris, Steve, and Andy Seaborne. 2013. "SPARQL 1.1 Query Language." *http://www.w3.org/TR/2013/REC-sparql11-query-20130321/*.

Harth, A, J Umbrich, A Hogan, and S Decker. 2007. "Yars2: a Federated Repository for Querying Graph Structured Data from the Web." *6th International Semantic Web Conference (ISWC)*, 211–224.

Hartig, O, C Bizer, and J C Freytag. 2009. "Executing SPARQL Queries over the Web of Linked Data." *8th International Semantic Web Conference ISWC2009*, 293–309.

Koutris, Paraschos, Prasang Upadhyaya, Magdalena Balazinska, Bill Howe, and Dan Suciu. 2013. "Toward Practical Query Pricing with QueryMarket." In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*.

Labrinidis, Alexandros, Huiming Qu, and Jie Xu. 2007. "Quality Contracts for Real-time Enterprises." In *Proceedings of the 1st international conference on Business intelligence for the real-time enterprises*. Berlin, Heidelberg:Springer-Verlag.

Lai, Kevin, Lars Rasmusson, Eytan Adar, Li Zhang, and Bernardo A. Huberman. 2005. "Tycoon: An Implementation of a Distributed, Market-based Resource Allocation System." *Multiagent and Grid Systems*, 1(3): 169–182.

Langegger, A, W Wöß, and M Blöchl. 2008. "A Semantic Web Middleware for Virtual Data Integration on the Web." *Lecture Notes In Computer Science*.

Lubin, Benjamin, and David C. Parkes. 2009. "Quantifying the Strategyproofness of Mechanisms via Metrics on Payoff Distributions." In *Proceedings of the 17th National Conference on Artificial Intelligence*.

Lubin, Benjamin, Benedikt Bünz, and Sven Seuken. 2015. "Fairness Beyond the Core: New Payment Rules for Combinatorial Auctions." Working Paper.

Malone, Thomas W., Richard E. Fikes, and Michael T. Howard. 1983. "Enterprise : a Market-like Task Scheduler for Distributed Computing Environments."

Milgrom, Paul. 2007. "Package Auctions and Exchanges." *Econometrica*, 75(4): 935–965.

Oren, E, C Gueret, and S Schlobach. 2008. "Anytime Query Answering in RDF through Evolutionary Algorithms." *The Semantic Web - ISWC 2008*, 5318: 98–113.

Parkes, David C., Jayant Kalagnanam, and Marta Eso. 2001. "Achieving Budget-balance with Vickrey-based Payment Schemes in Exchanges." In *Proceedings of the 17th International Joint Conference on Artificial Intelligence*. San Francisco, CA.

Quilitz, B, and U Leser. 2008. "Querying Distributed RDF Data Sources with SPARQL." *Proceedings of the 5th European semantic web conference on The semantic web: research and applications*, 524–538.

Schwarte, Andreas, Peter Haase, Katja Hose, Ralf Schenkel, and Michael Schmidt. 2011. "FedX: Optimization Techniques for Federated Query Processing on Linked Data." In *International Semantic Web Conference (1)*.

Stonebraker, Michael, Paul M. Aoki, Witold Litwin, Avi Pfeffer, Adam Sah, Jeff Sidell, Carl Staelin, and Andrew Yu. 1996. "Mariposa: A Wide-Area Distributed Database System." *VLDB J.*, 5(1): 48–63.

Sutherland, I. E. 1968. "A Futures Market in Computer Time." *Commun. ACM*, 11(6): 449–451.

Van Alstyne, Marshall, Erik Brynjolfsson, and Stuart Madnick. 1995. "Why Not One Big Database? Principles for Data Ownership." *Decis. Support Syst.*, 15(4): 267–284.

Waldspurger, C.A., T. Hogg, B.A. Huberman, J.O. Kephart, and W.S. Stornetta. 1992. "Spawn: a Distributed Computational Economy." *IEEE Transactions on Software Engineering*, 18(2): 103–117.

Zemánek, J, S Schenk, and V Svatek. 2007. "Optimizing SPARQL Queries Over Disparate RDF Data Sources Through Distributed Semi-joins." *7th International Semantic Web Conference (ISWC)*.