A BROKER FOR SELECTING AND PROVISIONING HIGH QUALITY SYNDICATED DATA

(Research Paper)

Danilo Ardagna, Cinzia Cappiello, Marco Comuzzi, Chiara Francalanci, Barbara Pernici
Politecnico di Milano, Milano, Italy
ardagna, cappiell, comuzzi, francala, pernici @elet.polimi.it

Abstract: Modern business strategies are focused on customer satisfaction and especially on one-to-one marketing. Customer preferences and needs are inferred by analyzing information on their behaviour and attitudes. This information can be collected, stored, and analyzed by implementing data warehouse and CRM functionalities. However, companies also need market information about their competitors. A solution to obtain this information is to request data from syndicated data providers. The market of syndicated data is heterogeneous. Data providers own different data sets characterized by different quality and granularity. To obtain the required information, customers must buy multiple data sets from different providers and then clean and merge them. This paper proposes a broker architecture that works as an intermediary between users and syndicated data providers. On the basis of data quality and cost requirements, the broker builds the most suitable data set by integrating data from different providers. In the selection phase, the broker uses optimization and negotiation mechanisms in order to satisfy requirements.

Key Words: syndicated data, data quality, quality optimization, quality negotiation.

1. INTRODUCTION

Modern business strategies are focused on customer satisfaction. Marketing is aimed at predicting customer preferences and designing products and services accordingly. Customer preferences and needs are inferred by analyzing information on their behaviour and attitudes. Data warehouses, data mining functionalities, and CRM applications are commonly adopted by large enterprises to collect, store, and analyze customer data. However, companies also need market information about their competitors. A solution to obtain this information is to request data from *syndicated data providers*.

Syndicated data providers play a key role in the modern information economy. Information can provide a competitive advantage and information quality is a determinant of the success of marketing initiatives. The market of syndicated data is heterogeneous and a major difficulty for enterprises is the selection of a data provider within a multitude of potential suppliers. Syndicated data are gathered with a variety of data collection mechanisms. These include surveys, questionnaires, polls, forums or transaction processing applications gathering data through various tracking devices. Usually, syndicated data providers are different from each other as they obtain data with different collection mechanisms and from different sources. Usually, a data source, such as a merchandising outlet, signs a data provisioning contract with a single provider. Therefore, companies are often forced to purchase data from different providers in order to have a complete picture of their customers' behaviour and attitudes.

When data can be obtained from multiple providers, companies often select their supplier inefficiently. Companies tend to interact with a limited set of data providers, which are often leader in the syndicated data market [8]. This excludes minor providers from the selection process, although niche players could outperform market leaders in the quality of specific data sets. This paper aims at supporting organizations in the supplier selection phase. We propose the use of an intermediary infrastructure, called *broker,* for the selection and provision of high quality syndicated data. On the basis of the requested data and of the quality and cost requirements associated with the request, the broker finds the most suitable solution by integrating information among different available data sets owned by different providers and suggesting the set of best data sets to the client. In order to guarantee the efficiency of the selection phase, the broker is based on optimization and negotiation mechanisms based on quality and cost parameters.

The paper is organized as follows. Section 2 presents the quality broker architecture, justifies the data quality dimensions that are considered in the paper and discusses the data model and the query mechanisms implemented by the broker. Section 3 focuses on the brokering methodology and presents all the actions that the broker performs in order to satisfy requests. Section 4 provides an example to clarify the selection process and outcome. Section 5 reviews alternative approaches in the literature. Conclusions are drawn in Section 6.

2. A QUALITY-ORIENTED REPRESENTATION OF SYNDICATED DATA

The broker is supposed to receive a query from a customer specifying the data request along with quality and cost requirements (Figure 1). The broker processes the query by selecting the most suitable set of data sets. The next section discusses the data model supporting the specification of both queries and responses.

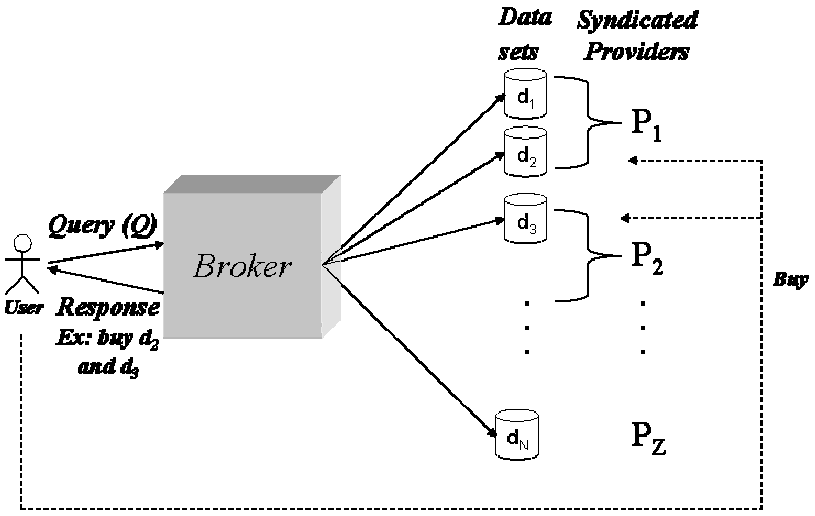


Figure 1- Broker architecture

2.1 The data model

The broker is modelled according to the Local-As-View (LAV) perspective [11]. Accordingly, the data of a provider are represented as views of a global schema, called broker schema. The broker knows both this global schema and how to build the global data set by integrating the data of all providers. Figure 2 shows the global data set. Data are divided into fragments f_{ek}, organized into E rows r_e and K columns c_k. $|f_{ek}|$ indicates the cardinality of data fragment f_{ek}, i.e. the number of data values contained in f_{ek}. The broker knows the cardinality of all data fragments.

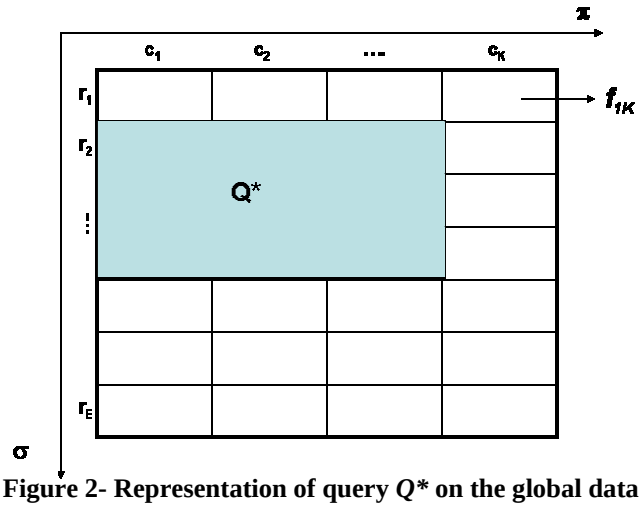


Figure 2- Representation of query *Q on the global data set GD**

Users specify:

- A query Q on the global schema representing the data set that is requested. Q is based on selection and projection operations. The projection operation selects a set of columns from the global schema, while the selection operation identifies the rows that satisfy the selection condition (see Figure 2).
- The data quality requirements that should be satisfied by the data set extracted by query Q. Data quality requirements are specified for different quality dimensions QD_r, e.g. accuracy, completeness, and timeliness, described in Section 2.2. Users specify the minimum level of quality that is considered acceptable, referred to as QD_r^*. Users can also express preferences among quality dimensions by specifying weights w_r, with $1 \ r \ R$, such that $\sum_{r=1}^{R} w_r = 1$. The weights identify the relevance associated with the data quality dimensions and the overall quality q is expressed as $q = \sum_{r=1}^{R} w_r QD_r$. A constraint q^* on the overall value of quality q is also specified.
- Price requirements in terms of the maximum price that they are willing to pay, *Price**.

On the basis of available data sets, the broker identifies the result Q* of query Q. Q* is a set of fragments, according to the global schema representation illustrated in Figure 2.

The data sets that can contribute to build the response Q^* are referred to as d_i (see Figure 3). Each data set represents the smallest subset of data that can be supplied by a provider including all the data values owned by the provider that satisfy Q. We assume that data quality is homogeneous within each data set (*HP1*).

A data set d_i is defined as a set of fragments. For each d_i, we define the number of its elements $|d_i|$ as the sum of the cardinalities of all fragments in d_i. In general, the query result Q^* can be built by using multiple combinations of d_i. The broker identifies all data sets that can contribute to satisfy the user's request, either globally or partially. A *query plan* is defined as a vector $\vec{x} = \langle x_1, x_2 \ldots x_N \rangle$ such that:

$$x_i = \begin{cases} 1 & (\mathrm{d_i} \cap Q^*) \neq 0; \quad \text{i.e, data set } \mathrm{d_i} \text{ can contribute to build } Q^* \\ 0 & \text{otherwise} \end{cases}$$

Let $D = \{d_i \mid i = 1,...,N\}$ be the set of all data sets, where N represents the total number of data sets available in the system. The size of the solution domain has an order of magnitude equal to 2^N-1. The goal of the optimization algorithm presented in Section 3.2 is to identify the optimum query plan $\vec{x}^*$ without exploring all possible solutions.

Query plans are identified by considering all data sets d_i available in the system (see Figure 3). As represented in Figure 4, data sets can overlap. With od_u, we denote the overlaps among data sets d_i.

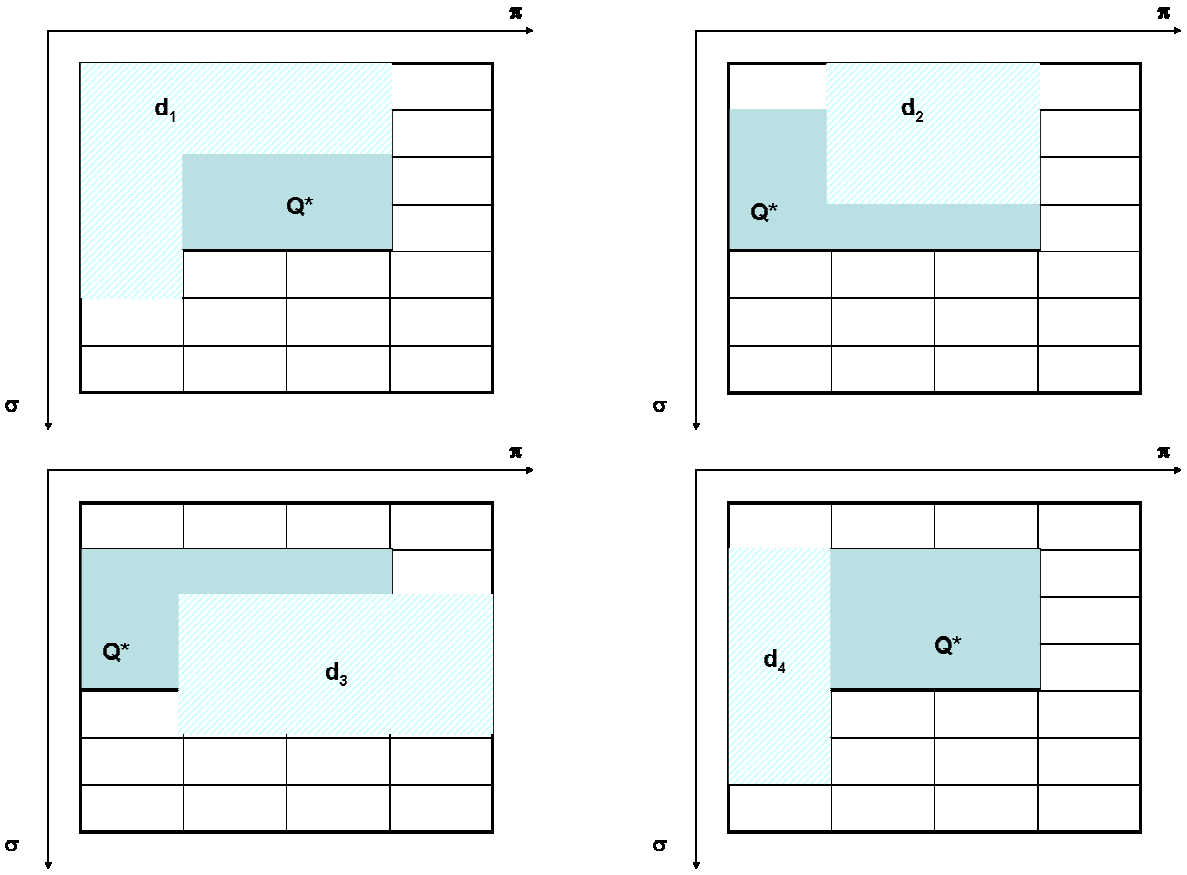


Figure 3 – Different data sets including data satisfying Q*

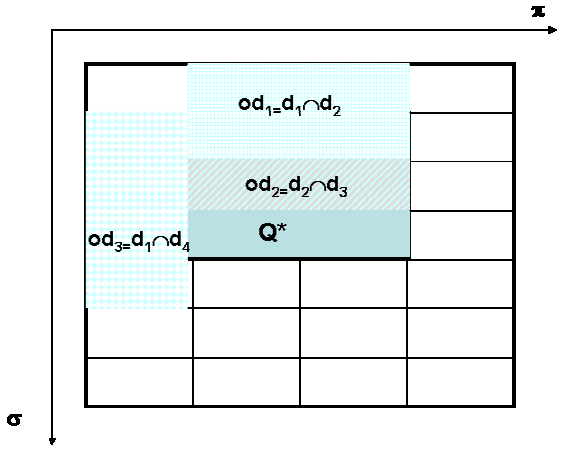


Figure 4 - Overlaps among data sets

The broker is in charge of managing the relationship with providers, but has no visibility on data values. The broker is also supposed to receive the average value of quality $QD_r(d_i)$ of each data set from corresponding providers. We assume that providers have the responsibility for the evaluation of data quality, i.e. each provider has implemented data quality tools for the evaluation of data quality along the dimensions considered in this paper.

2.2 Data Quality Dimensions

The quality dimensions considered in this paper are accuracy, completeness, and timeliness. However, our model can be extended to consider other quality dimensions. Accuracy and completeness assess data along their numerical extension and correctness. Timeliness evaluates the validity of data along time. Accuracy, completeness, and timeliness are objective dimensions and, therefore, are suitable for a quantitative evaluation. These three dimensions constitute a minimal set that provides sufficient information about the suitability of data along the process in which they are involved. In the following we clarify the definition associated with each data quality dimension.

Different definitions of accuracy are provided in the data quality literature [15][18]. In the following, we adopt a measure of accuracy associated with data sources, defined as the ratio between the number of correct values and the total number of values available from a given source [15]. This definition supports the mathematical operations that involve the accuracy dimension discussed in Section 3.1.

The definition of completeness is consistent across research contributions. In [15] completeness is associated with data values and is defined as the degree to which a specific database includes all the values corresponding to a complete representation of a given set of real world events as database entities.

Timeliness is defined as the property of information to arrive early or at the right time [5]. We measure timeliness as a function of two elementary variables, currency and volatility [2][3]. The measure of timeliness is defined as in [2]:

$$Timeliness = \max\left(1 - \frac{Currency}{Volatility}; 0\right)^s,$$

where exponent s is a parameter necessary to control the sensitivity of timeliness to the currency-volatility ratio. With this definition, the value of timeliness ranges between 0 and 1.

Currency is not provided a standard definition in the literature. Currency is usually defined as a time measure [2][3][18]. In this paper we use the definition provided in [3], in which currency is defined as the time interval that goes from the time when data are updated to the time when data are used. Volatility, is the time interval measured as the average time length for which data remain valid [2]. Volatility is considered a static property that is dependent of the frequency of updates.

Note that currency and, consequently, timeliness, have an influence on both accuracy and completeness. For example, delays in propagating changes across databases are a cause for either inaccuracy or incompleteness [4]. If new data are created in a database, other databases may be incomplete until changes are propagated. If existing data are updated, other databases are inaccurate until propagation. The higher the number of changes, the lower the accuracy and completeness of data.

With these definitions, accuracy, completeness, and timeliness are *positive* quality dimensions, i.e. the higher is their value, the higher the quality perceived by the end user. This characteristic allows us to use the generic notation QD_r to refer to any quality dimension in the mathematical model presented in Section 2.3 and to introduce “greater or equal to” constraints.

2.3 Evaluation of the quality of data from multiple sources

To support the selection of the most suitable set of suppliers, the broker must calculate the overall quality of each query plan. We assume that $QD_r(d_i)$ is normally distributed among data fragments f_{ek} contained in data set d_i (*HP1*). The quality of a query plan is indicated as $QD_r\left(\vec{x}\right)$. In order to calculate $QD_r\left(\vec{x}\right)$, let us consider the set V of overlapping data sets. In our model, we define an overlap among data sets as:

$$od_u = \bigcap_{d_i \in OD_u} d_i$$

Where OD_u is a member of the power set 2^V such that $|OD_u| > 1$ and u varies between 1 and $U=2^{|V|}-1-|V|$. By definition, fragments belonging to Q^* will be covered by at least one data set.

a) If V is the empty set, i.e. there are no overlaps among d_i, then:

$$QD_r(\vec{x}) = \frac{\sum_{i=1}^{N} QD_r(d_i) \cdot |d_i \cap Q^*| \cdot x_i}{|Q^*|}$$

Where $|d_i \ Q^*|$ represents the number of data values (i.e., the cardinality multiplied by the number of attributes) that data set d_i provides to build query result Q^*.

b) If V is different from the empty set, i.e. there are overlaps among d_i, then for each $od_u \neq$ it is necessary to identify the set d_i' that is a subset of d_i which does not overlap with any other data set:

$$d_i' = d_i \setminus \bigcup_{u=1}^{U} od_u$$

The $QD_r(d_i')$ value is calculated from $QD_r(d_i)$ on the basis of *HP1* of uniform distribution.

$$QD_r(d_i') = \frac{QD_r(d_i \cap Q^*)}{|d_i \cap Q^*|} \cdot |d_i' \cap Q^*|$$

The quality dimension evaluation of the overlapped parts is determined as follows, considering the data sets associated with the best data quality. For each od_u, we select the data set $d_i^* \ OD_u$ which maximizes the overall quality $q = \sum_{r=1}^{R} w_r \cdot QD_r(d_i)$.

Next, we identify the set od_u' that does not overlap with other overlapping data sets:

$$od_u' = od_u \setminus \bigcup_{\substack{v=1 \\ v \neq u}}^{U} od_v$$

Then $QD_r(od_u)$ is given by:

$$QD_r(od_u) = \frac{QD_r(d_i^*)}{|od_u|} * |od_u'|$$

c) Once that $i,u \ QD_r(d_i)$ and $QD_r(od_u)$ have been calculated, $QD_r(\vec{x})$ is given by:

$$QD_r(\vec{x}) = \frac{\sum_{i=1}^{N} x_i \cdot QD_r(d_i') \cdot |d_i'| + \sum_{u=1}^{U} QD_r(od_u') \cdot |od_u'|}{|Q^*|}$$

d) $QD_r(\vec{x})$ provides an aggregate value of quality dimensions QD_r for query plan $\vec{x}$. Since we analyze an aggregate entity, the same value can correspond to different distributions of the value QD_r on the data sets. For example, let us compare a query plan $(\vec{x})_1$ in which all the data sets are characterized by an average quality with another query plan $(\vec{x})_2$ in which high quality data sets are alternated with very low quality data sets. $(\vec{x})_1$ and $(\vec{x})_2$ can be associated with the same aggregate value QD_r and in a preliminary analysis they will be considered as equivalent solutions. It is instead clear that for the users the query plan $(\vec{x})_1$ is a better solution than $(\vec{x})_2$. In order to avoid this critical situation and to provide to the users the most suitable solution along their needs, we introduce a new property called *uniformity*. Uniformity is defined as the degree with which a data quality dimension value QD_r varies in the data sets that compose a query plan $\vec{x}$. The measure of the uniformity is given by:

$$Uniformity_r(\vec{x}) = \sqrt{\frac{\sum_{i=1}^{N} |d_i'| (x_i \cdot QD_r(d_i') - QD_r(\vec{x}))^2 + \sum_{u=1}^{U} |od_u'| (QD_r(od_u') - QD_r(\vec{x}))^2}{|Q^*|}}$$

In order to be considered, a query plan has to be characterized by a uniformity value lower than a specified value $Uniformity_r^*$. In case two or more query plans are characterized by similar values of the quality dimension QD_r, it is preferable to select the query plan that is associated with the lowest uniformity value(see Section 3.1).

e) The broker calculates Price $(\vec{x})$, the price of the solution as:

$$\text{Price}(\vec{x}) = \sum_{i=1}^{N} Price(d_i) x_i$$

3. A DATA QUALITY BROKERING METHODOLOGY

A query plan is considered *feasible* if it satisfies both quality and price constraints. A query plan is optimum if it is feasible and maximizes quality. The goal of the broker is to select the optimum query plan $\vec{x}^*$. If no feasible plans exist, the broker can negotiate data quality characteristics with syndicated data providers. Providers can improve the quality of their data with an additional cost. Negotiation identifies a new set of candidate plans which may provide a solution satisfying constraints (see Figure 5). Our optimization algorithm is based on the tabu search approach, while the negotiation process is based on multi-party, multi-attribute, single-encounter negotiation. The next section discusses the optimization model. The optimization algorithm is presented in Section 3.2. The negotiation process is explained in Section 3.3.

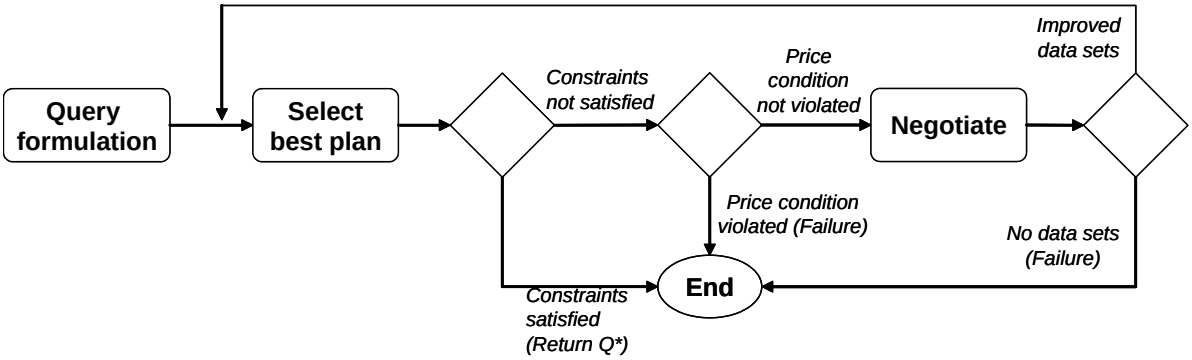


Figure 5 – Data quality brokering methodology

3.1 Formulation of the optimization problem

The identification of the optimum query plan $\vec{x}*$ can be formalized as follows:

P1)
$$\min \sum_{i=1}^{N} Price(d_i)x_i$$

$$QD_r(\vec{x}) \geq QD_r*, \forall r \in [1,R] \quad (1)$$

$$q(\vec{x}) \geq q* \quad (2)$$

$$Uniformity_r(\vec{x}) \leq Uniformity_r*, \forall r \in [1,R] \quad (3)$$

$$x_i \in [0,1]$$

We consider the problem of minimization of the query plan price with quality constraints instead of the dual problem of maximization of quality with a price constraint.

In this ways, if P1) has no feasible solution, the optimization domain can be extended by the negotiation process. Vice versa, if the dual problem has no feasible solution (i.e., all of the possible combinations of the data sets have a price greater than Price*), the negotiation process is not effective, since syndicated data providers can only improve the data quality by performing data cleaning procedures and increasing data sets prices.

Note that, if the price of the optimum solution of P1) is greater than Price*, then query Q cannot be satisfied, since no combinations of data sets have a price lower than Price* while satisfying quality constraints.

Problem P1) is an integer problem with a linear objective function and non linear constraints (constraint families (1)-(3)). If the end user does not specify a completeness constraint, the following constraint is introduced:

$$\bigcup_{i|x_i=1} d_i \supseteq Q* \quad (4)$$

which guarantees that the selected data sets cover the result Q* (see Figure 3).

Furthermore, constraint (2) can be strengthened as follows:

$$q(\vec{x}) \geq \max\left(q*, \sum_{r=1}^{R} w_r QD_r* \right) \quad (2')$$

constraint (2') can be obtained by adding up each constraint (1) multiplied by the corresponding weight w_r. Problem P1) is NP-hard, since it represents an extension of a set covering problem (see [14]).

3.2 Optimization algorithm

Our optimization approach is based on the tabu search (TS) algorithm [9]. Tabu-search is a meta-heuristic that guides a local search procedure to explore the solution space beyond local optimality. Let *X* denote the set of feasible solutions of our problem. To each $\vec{x}$ *X*, TS associates a subset of *X*, called neighborhood of $\vec{x}$ and denoted with $N(\vec{x})$. The neighborhood of $\vec{x}$ contains all the solutions that can be obtained with simple modifications of $\vec{x}$, called moves. Given a feasible solution $\vec{x}$, TS selects the solution with the best value of the objective function within a subset of $N(\vec{x})$. The selection of a solution in $N(\vec{x})$ is forbidden if that solution has already been selected in a previous iteration. Forbidden solutions are called tabu. To identify a solution, TS records the necessary information, called attributes, in a memory structure, called *tabu list*. The length of the tabu list, called *tabu-tenure T*, is limited and the tabu list is managed with a first in first out policy. The tabu status of a solution can be *overruled* if certain conditions, called *aspiration criteria*, are verified. A common adopted aspiration criterion accepts a tabu solution if its objective function is strictly better than that of all the solutions that have already been explored. TS stop criteria are based on total elapsed time or total number of iterations.

The query optimization algorithm is reported in Figure 6. First an initial solution $\vec{x}$ is built (step 1 in Figure 6). If the initial solution is not feasible, then $\vec{x}$ is modified by the *FindFeasibleSolution* procedure (step 3 in Figure 6), which may start the negotiation process. If $\vec{x}$ is still unfeasible, then the algorithm returns a null solution (step 5 in Figure 6). If $\vec{x}$ is feasible, the *TabuSearch* optimization procedure is performed and the query result *Q** is computed by the *Query* procedure (step 8 in Figure 6).
In the next section, the procedure *FindInitialSolution* is presented. In Section 3.2.2 we introduce the moves that implement the *TabuSearch* optimization procedure. The same moves are adopted by the *FindFeasibleSolution* procedure, which is discussed in Section 3.2.3.

```
1.  x ←FindInitialSolution();
2.  if x is unfeasible then
3.                  x ←FindFeasibleSolution( x );
4.  if x is unfeasible then
5.                  Q*←  ;
6.                  else{
7.                  x ←TabuSearch( x );
8.                  Q*←Query( x );
9.                  }
10. return Q*;
```

Figure 6 – Query optimization algorithm

3.2.1 Building an initial solution

The goal of the *FindInitialSlution* procedure is to find a solution that covers the result *Q**. First, data sets d_i are sorted by non increasing value of $\frac{\sum_r w_r QD_r(d_i)}{Price(d_i)} |d_i \cap Q^*|$. A data set is added to the initial solution following this order if the completeness (or the coverage of *Q**) is improved until the completeness constraint (or constraint (4)) is satisfied (or all data sets are added to the initial solution). This method favours data sets with a higher quality to cost ratio or with greater size. The complexity of the procedure is given by the sorting algorithm and is *O(N log N)*.

3.2.2 Tabu search optimization

The neighbourhood of a solution is defined by the following moves:

- add a data set d_i to the current solution, i.e. set $x_i=1$,
- remove a data set d_i from the current solution, i.e., set $x_i=0$,

Quality constraints (1)-(3) have to be evaluated in the exploration of the neighbourhood. Note that the overlapping data sets can be obtained from the current od_u without re-computing the whole power set 2^V (see Section 2.3). The search is guided by a tabu-search meta-heuristic in which only the short-term memory mechanism has been implemented [9]. The tabu list is implemented as a vector whose elements tl_i store the latest iteration that has updated decision variable x_i. Let l be the current iteration of the tabu search. A move is considered tabu if $l\text{-}tl_i \leq T$, where T represents the maximum length of the tabu list. The complexity of the neighbourhood exploration is O(N).

3.2.3 Finding a feasible solution

The TS algorithm requires an initial feasible solution which is the starting point of the neighborhood exploration. If the initial solution identified by the greedy algorithm discussed in Section 3.2.1 is unfeasible, then the procedure *FindFeasibleSolution()* reported in Figure 7 is executed.

1. $n \leftarrow 1$;
2. *CONTINUE*←TRUE;
3. **While** ($n \leq n_{Max}$) and *CONTINUE*{
4. $m \leftarrow 1$;
5. **While** ($m \leq m_{Max}$) and ($\vec{x}$ is unfeasible) {
6. Identify the most violated constraint $k(\vec{x}) \geq k^*$ among (1)-(4').
7. $N(\vec{x}) \leftarrow$EvaluateNeighborhood($\vec{x}$);
8. Identify $\vec{x}'$ $N(\vec{x})$ such that $\frac{k(\vec{x})-k^*}{k^*}$ is maximized and no constraint is violated;
9. $\vec{x} \leftarrow \vec{x}'$;
10. $m \leftarrow m+1$;
11. }
12. **if** ($\vec{x}$ is unfeasible) and ($Price(\vec{x}) \leq Price^*$) **then**
13. $D' \leftarrow$Negotiate($\vec{x}$);
14. $D \leftarrow D' \cup D$;
15. *CONTINUE*←TRUE;
16. $n \leftarrow n+1$;
17. **else**{
18. $\vec{x} = \mathbf{0}$
19. *CONTINUE*←FALSE;
20. }
21. }
22. return $\vec{x}$;

Figure 7 – *FindFeasibleSolution* procedure

At each iteration, the most violated constraint in percentage among (1)-(4') is identified $k(\vec{x})$. Then, the neighborhood introduced in Section 3.2.1 is explored and the move that allows the highest percent improvement of the constraint (without causing other violations) is selected (steps 4-9 in Figure 7). If a feasible solution cannot be found, but the price of the current solution is lower than the constraint $Price^*$ (step 10), then the negotiation procedure is executed, higher quality data sets are obtained (step 13) and the neighborhood exploration is restarted in a broader solution domain (steps 14-16). The algorithm is repeated until a feasible solution is found or the maximum number of iterations is reached.

Note that, if the current solution has a price higher than *Price**, i.e. both quality and price constraints are violated, then we argue that no feasible solution exists. The search is stopped (step 18) and the broker returns an empty set as result of *Q**.

3.3 The negotiation process

According to Figure 7, negotiation is required when a solution violates data quality constraints, while satisfying the price constraint. The goal of the negotiation process is to generate new set of data fragments *D'* with higher quality. Our negotiation algorithm is an adaptation of the service-oriented algorithm described in [7]. Two types of negotiation are started depending on constraint violations:

- *Case 1*: one among constraints 1-3 is the most violated. In this case, negotiation will focus on price and on the most violated data quality dimension.
- *Case 2:* constraint 4 is the most violated; in this case, negotiation will focus on price and on all quality dimensions, i.e., completeness, accuracy and timeliness.

The negotiation process can be always considered multiparty, multiattribute, and single encounter. Multiple parties are involved, i.e., all providers supplying a data set d_i in $\vec{x}$ and the broker. Negotiation is multiattribute since at least price and a data quality dimension are negotiated. Finally, negotiation is single encounter, since each broker-provider negotiation is considered as an independent bilateral bargaining problem. Thus, the whole negotiation process is defined as a set of parallel bilateral bargaining sessions between the broker and each provider.

The specification of an automated negotiation process relies on three elements [10]:

- *Negotiation objectives:* they define the features of the object or service that is negotiated.
- *Negotiation protocol:* it identifies the participants involved in the negotiation process and the actual protocol adopted in the process, in terms of allowed messages and message flow patterns between the parties.
- *Decision models of negotiation parties:* they define the behaviour of negotiation parties in terms of strategies to produce offers and counter-offers, methods to evaluate offers, and rules to determine whether an offer cab or cannot be accepted.

Case 1 and 2 described above provide our negotiation objectives. The messages exchanged by negotiation parties and their decision models are formalized in the next section.

3.3.1 Decision models of broker and providers

First, it is important to identify the value ranges of negotiation attributes. According to the outcome of the optimization process, the difference between the maximum price *Price** and the price associated to the unfeasible query plan $\vec{x}$ represents a price gap that the broker can exploit during negotiation. This gap is partitioned among the providers that contribute to solution $\vec{x}$. The fraction of price gap that is allocated to each provider is proportional to the fraction of data provided to build *Q**. Therefore, the price gap $PG(d_i)$ considered for the negotiation on data set d_i is defined as:

$$PG(d_i) = (Price^* - Price(\vec{x})) \cdot \frac{\left| d_i \cap Q^* \right|}{\left| Q^* \right|}.$$

In *Case 1*, a single quality dimension QD_r is negotiated. Let us consider the *accuracy* dimension and define the variable QD_1 as *Acc*. When negotiating with the z^{th} provider, the broker has to fill a gap between the objective value of accuracy *Acc** and the actual value $Acc(\vec{x})$associated with the unfeasible starting solution. In *Case 2*, the gap is identified by the aggregate quality value *q** and the corresponding aggregate value $q(\vec{x})$ of the starting solution. The gap is partitioned among the providers according to

their contribution to Q^* and among quality dimensions according to weights w_r. In *Case 1*, the gap allocated to data set d_i is defined for quality dimension QD_r as follows:

$$QDG_r(d_i) = \left[QD_r^* - QD_r(\vec{x})\right] \cdot \frac{\left|d_i \cap Q^*\right|}{\left|Q^*\right|}.$$

In *Case 2,* for each quality dimension the broker allocates a quality gap that is weighed according to the importance attributed by the user to each quality dimension:

$$QDG_r(d_i) = \left[q^* - q(\vec{x})\right] \cdot \frac{\left|d_i \cap Q^*\right|}{\left|Q^*\right|} \cdot w_r .$$

Negotiation ranges are shown in Table 1. The upper bound of each quality dimension is related to the providers' ability to perform data cleaning and increase quality, while the upper bound of price is the price associated with the data cleaning activity by the provider ($P_{incr}(d_i)$). By adopting a quadratic model of the cost of data cleaning (see [6]), the new price $Price(d_i')$ is evaluated as:

$$\text{Price}(d'_i) = \text{Price}(d_i) \cdot \left(1 + \sum_{r=1}^{R}\left[\alpha_r^z \cdot \left(QD_r(d_i') - QD_r(d_i)\right)\right]^2\right)$$

where α_r^z are parameters describing the z^{th} provider.

	Broker	**Provider**
Price Range	$\left[Price(d_i), Price(d_i) + PG(d_i)\right]$	$\left[Price(d_i), Price^{MAX}(d_i)\right]$ $Price^{MAX}(d_i) = Price(d_i) + Price_{incr}(d_i)$
Generic data quality dimension QD_r Range	$\left[QD_r(d_i), QD_r(d_i) + QDG_r(d_i)\right]$	$\left[QD_r(d_i), QD_r^{MAX}(d_i)\right]$

Table 1-Definition of negotiation ranges

In the remainder, price and quality values are considered generic negotiation attributes a_h, with *h=1,2* in *Case 1* and *h=1,…,R+1* in *Case 2. H* is the number of negotiated attributes. We refer to $a_h^{\min}$ as the lower bound of a generic negotiation attribute *h*; conversely, $a_h^{\max}$ is the upper bound (e.g., $Price(d_i)+PM(d_i)$ for the broker and $Price_{MAX}(d_i)$ for the provider). During the negotiation process, an offer is represented as a vector $\vec{A}$ of values a_h. In *Case 1*, $A=(a_1,a_2)$, *h=1* identifies price and *h=2* the most violated quality dimension; in *Case 2*, $A=(a_1,a_2,\ldots,a_H)$, where a_1 is price.

The decision model of negotiation parties requires the specification of utility functions which are used by each party to evaluate their counterpart's offer and make an accept of reject decision. In this paper, we consider a global utility function defined as the weighed sum of utility functions defined for each negotiated attribute:

$$V^B\left(\vec{A}\right) = \sum_{h=1}^{H} \omega_h^B \cdot V_h^B\left(a_h\right), \text{ for the broker B,}$$

$$V^z\left(\vec{A}\right) = \sum_{h=1}^{H} \omega_h^z \cdot V_h^z\left(a_h\right), \text{ for the } z^{th} \text{ provider,}$$

where $\sum_{h=1}^{H} \omega_h = 1$, for both the broker *B* and the z^{th} provider. The z^{th} provider sets the value of weights ω_h^z according to its own business strategies. The values of ω_h^B are calculated as follows. From the broker's perspective, price becomes an important negotiation attribute when the price gap is low. Once the weight associated with the price is evaluated, weights related to data quality dimensions can be directly derived from the *w* weights, already defined to measure the user preferences along different quality dimensions. Thus, $\omega_1^B = 1 - \frac{PG(d_i)}{Price^*}$, in every case. In *Case 1*, $\omega_2^B = 1 - \omega_1^B$, while in *Case 2*, $\omega_h^B = \left(1 - \omega_1^B\right) \cdot w_r$. In the latter case, it is easy to verify that ω_h^B add up to 1:

$$\sum_{h=1}^{H} \omega_h^B = \omega_1^B + \sum_{h=2}^{H} \left(1 - \omega_1^B\right) \cdot w_h = \omega_1^B + \left(1 - \omega_1^B\right) \sum_{r=1}^{R} w_r = \omega_1^B + \left(1 - \omega_1^B\right) = 1$$

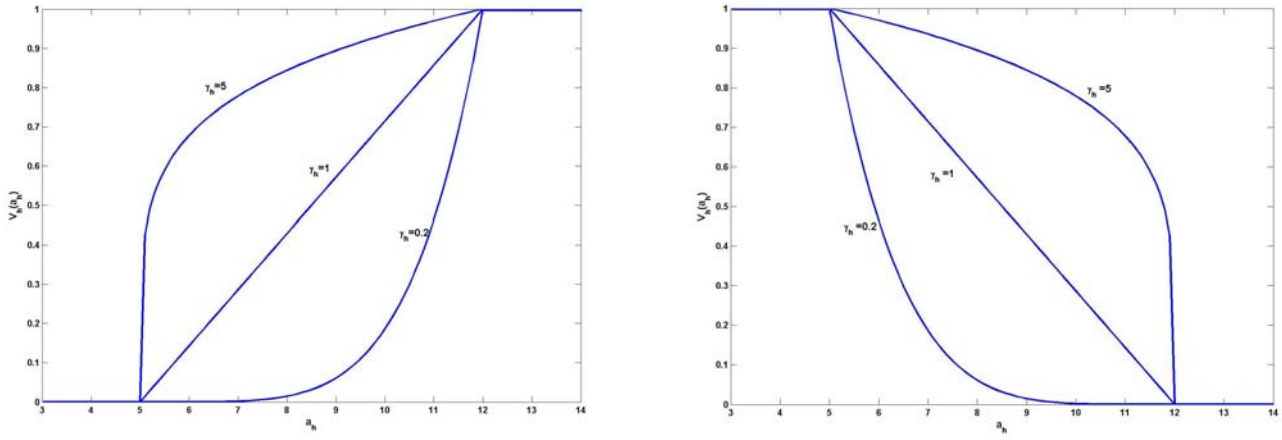
Functions $V_h(a_h)$ can assume two different forms. When utility increases as a consequence of an increase of a negotiation attribute, $V_h(a_h)$ is expressed as:

$$V_k(a_h) = \begin{cases} 0 & a_h \le a_h^{\min} \\ \left[\left(a_h - a_h^{\min}\right) \Big/ \left(a_h^{\max} - a_h^{\min}\right)\right]^{1/\gamma_h} & a_h^{\min} < a_h < a_h^{\max} \\ 1 & a_h \ge a_h^{\max} \end{cases}.$$

For instance, the broker attributes greater utility to higher values of data quality dimensions and, similarly, the provider attributes greater utility to a higher price. On the contrary, when the utility decreases as a consequence of an increase of a negotiation attribute, such as price for the broker, $V_h(a_h)$ is expressed as:

$$V_k(a_h) = \begin{cases} 1 & a_h \le a_h^{\min} \\ \left[1 - \left(a_h - a_h^{\min}\right) \Big/ \left(a_h^{\max} - a_h^{\min}\right)\right]^{1/\gamma_h} & a_h^{\min} < a_h < a_h^{\max} \\ 0 & a_h \ge a_h^{\max} \end{cases}.$$

The γ_h parameter controls the trend of the $V_h(a_h)$ function, as shown in Figure 8. In case of increasing utility values, for lower values of γ_h negotiation parties are satisfied only if the offer to be evaluated is close to the upper bound associated with the negotiation parameter. Obviously, utility remains high even if the negotiation parameter exceeds its upper bound.



(a) *V* increasing

(b) *V* decreasing

Figure 8 – Sample utility functions (a_h^{min}=5 and a_h^{max}=12)

In addition to utility functions, the decision model of negotiation parties is defined by their strategies, that is the set of rules applied to generate offers and counter-offers. For the sake of clarity, it is useful to describe the typical scenario of a bilateral bargaining between the broker and the z^{th} provider. Let us suppose that the broker makes the first offer, i.e., the broker proposes a vector of values $\vec{A}$ to the provider. A global deadline t_{max} is associated with each bargaining process. Time instants are associated with offers and counter-offers: the first offer is posted by the broker at time *t=0*, the provider's counter-offer at *t=1*, the broker replies at *t=2*, and so on. The process ends when a participant accept the last offer made by the counterpart or when time exceeds the deadline t_{max}. In this latter case, an agreement is not reached by negotiation parties and the broker cannot introduce a new data fragment in *D'*.

Let us consider negotiation at time *t*. Let us suppose that the broker has just sent an offer $\vec{A}_{t-1}$ to the provider. The provider defines counter-offer A_t according to its own strategy and accepts the broker's proposal $\vec{A}_{t-1}$ if the following condition is verified:

$$V^z\left(\vec{A}_{t-1}\right) \geq V^z\left(\vec{A}_t\right),$$

otherwise it will post counteroffer A_t.

The evaluation of each term a_h^t in A_t is different when considering attributes with increasing or decreasing utility values, in particular:

$$a_h^t = a_h^{\min} + \left(1 - g_h^z\left(t\right)\right) \cdot \left(a_h^{\max} - a_h^{\min}\right)$$

if utility is increasing, and:

$$a_h^t = a_h^{\min} + g_h^z(t) \cdot \left(a_h^{\max} - a_h^{\min}\right)$$

if utility is decreasing.

The function $g_h^z(t)$ measures the time-dependent degree of concession of the provider on the h^{th} negotiation attribute and is expressed as:

$$g_h^z\left(t\right) = k_h^z + (1 - k_h^z) \cdot \left({}^{t}\!/_{t_{\max}} \right)^{1/\beta_h^z}$$

The trend of $g_h^z(t)$ for different values of the β_h^z parameter is shown in Figure 9. Higher values of β_h^z are associated with a more conceding behaviour, since the provider will move faster towards values of the

negotiation attribute that are closer to the broker's offer, while lower values of β_h^z are associated with a non conceding behaviour, generally referred to as *boulware* in the classical negotiation literature [10]. Similar functions $g_h^B(t)$ are defined to characterize the broker's strategy.

The broker's first offer is assembled by considering upper bounds for negotiation parameters with increasing utility values and lower bounds for negotiation parameters with decreasing utility. According to this strategy, the utility value associated with the first offer will be maximum. Similarly, if the provider posts the first proposal, it will use its own upper and lower bounds of negotiation attributes.

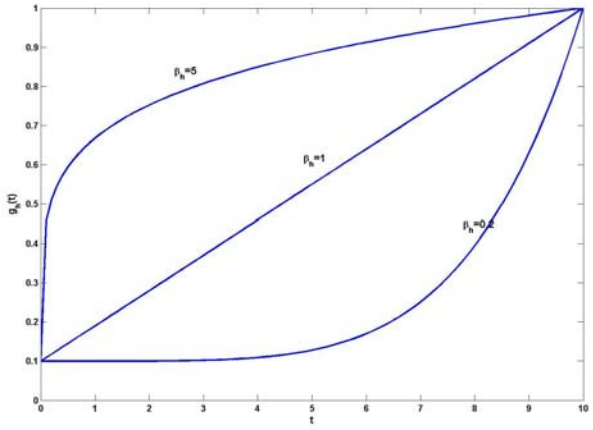


Figure 9- $g_h(t)$ for $k_h=0.1$ and $t_{max}=10$

In summary, the provider's decision model is fully specified by assigning a value to:

- ω_h^z, for the global utility function;
- γ_h^z, for the utility functions defined for individual negotiation attributes;
- β_h^z, defining the degree of concession on each negotiation attribute.

The broker's decision model is specified by:

- γ_h^B, for the utility functions defined for individual negotiation attributes;
- β_h^B, defining the degree of concession on each negotiation attribute.

4. A RUNNING EXAMPLE

In this section, we present a simple test case showing how optimization and negotiation are interleaved in order to identify the final query plan.

The end user requires data on customers which access financial services by means of mobile phones. Let us consider the following query *Q*: provide the number of banking and trading transactions and the corresponding average value made by customers whose age is in the range 30-50 years, such that the maximum price is 8.000$, while completeness and accuracy are greater than or equal to 0.9 and 0.8, respectively. Let us assume that weights w_1 and w_2 for completeness and accuracy are equal to 0.5. Query result *Q** and the available data sets provided by four different syndicated data providers are reported in Figures 3 and 4, where:

- d_1 includes banking and trading transaction data for customers less than 40 years old;
- d_2 provides data on banking transactions for customers less than 45 years old;
- d_3 provides data on trading transactions for customers living in the United Kingdom;
- d_4 includes data on trading transactions.

The price, completeness and accuracy of data sets are reported in Table 2, while the characteristics of overlapping data sets are reported in Table 3. Let us assume that $|Q^*|=900$. In the providers' pricing models we set $\alpha_r^z = 10$ for each QD_r and for each provider z.

Data set	Price	Completeness	Accuracy	$\left\|d_i \cap Q^*\right\|$	$\frac{\sum_r w_r QD_r(d_i)}{Price(d_i)}\left\|d_i \cap Q^*\right\|$
d_1	2.000 $	0.8	0.7	500	0.375
d_2	2.200 $	0.85	0.85	400	0.309091
d_3	2.500 $	0.96	0.85	400	0.2896
d_4	3.000 $	0.75	0.8	300	0.155

Table 2 – Characteristics of data sets

Overlapping data set	Completeness	Accuracy	$\left\|od_u \cap Q^*\right\|$
od_1	0.85	0.85	200
od_2	0.95	0.85	200
od_3	0.75	0.8	300

Table 3 – Quality values of overlapping data sets

The initial solution includes data sets d_1, d_2, and d_3 with a $ 7.000 total price and with completeness and accuracy equal to 0.88 and 0.78, respectively. Both completeness and accuracy constraints are not fulfilled, and accuracy is the most violated constraint. The initial solution is improved by the *FindFeasibleSolution()* procedure which adds d_4 and removes d_2 from the initial plan. In this way, the accuracy constraint is satisfied while the completeness constraint remains violated (the plan $\vec{x}$=<1,0,1,1> has completeness 0.854 and accuracy 0.8). Then, the negotiation process is started. Negotiation parameters are reported in Table 4. Negotiation with P_1 and P_4 leads to two new data sets d_5 and d_6 which replace d_1 and d_4 in the current plan $\vec{x}$=<0,0,1,0,1,1> and are characterized by the attributes reported in Table 5.

	β_{Price}	$\beta_{\text{Completeness}}$	$\gamma_{price,completeness}$
Broker	4	1	1
P_1	3.3	3	1
P_3	0.05	0.05	1
P_4	14	0.5	1

Table 4 – Negotiation parameters

Data set	Price	Completeness	Accuracy
d_5	2.250 $	0.87	0.7
d_6	3.100 $	0.92	0.8

Table 5 – Characteristics of data sets d_5 and d_6

The new plan is feasible since completeness evaluates to 0.92 and total price is 7.850$. The optimization process is started and the TS algorithm terminates by providing the optimum solution $\vec{x}$*=<0,1,1,0,0,1>, i.e. d_5 is replaced by d_2, with a total price equal to $ 7.800. Completeness and accuracy are equal to 0.92 and 0.83, respectively.

5. RELATED WORK

Architectures for the data quality management have been designed in order to evaluate and improve data. In the particular context of Cooperative Information Systems (CIS), a Data Quality Broker has been proposed for the selection of the best data sources satisfying quality requirements [16]. The broker receives a user request and sends corresponding data requests to the organizations belonging to the CIS. The broker is based on a GAV (Global As View) approach, since it is responsible for data retrieval and reconciliation. Reconciliation is performed by choosing the data values characterized by highest quality. The paper does not provide a mathematical model for the calculation of overall quality and does not consider price. The optimization of the query plan has been addressed in [1], [12], and [13]. Authors in [1] have considered a linear formulation of P1) which is obtained by considering a priori all possible intersections of overlapping data sets and by pre-computing corresponding quality values. The problem is solved by state of the art integer linear solvers and the optimum solution is identified. However, their formulation is not appropriate for our objectives. If no feasible solution exists, then the negotiation process identifies a new set of candidate solutions with different quality characteristics. In this way, after a negotiation process, the number of data sets N increases, the number of overlapping data sets V also increases (see Section 2.3) and corresponding quality values have to be re-evaluated. Since the number of evaluations ($U=2^{|V|}-1-|V|$) grows exponentially with the size of the problem, the approach proposed in [1] has a limited scalability. Vice versa, our neighborhood exploration considers data overlaps among the data sets of the current solution only and, as discussed in Section 3.2.2, the quality value of a new feasible solution can be evaluated from the current one. Authors in [12] and [13] consider the maximization of data quality with no price constraints. They formulate a non linear problem which is solved by implementing a branch and bound algorithm. Since no price constraint is introduced, they can also exclude low quality data sets a priori, which leads to sub-optimal solutions. With our price constraint, low quality data sets cannot be excluded a priori and the trade off between quality and price must be evaluated. Very high quality data sets could lead to unfeasible solutions, while low quality data sets could provide useful data. The branch and bound algorithm identifies the optimum solution of the problem, but the worst case execution time grows exponentially with the number of nodes of the underlying decision tree [20], which is obtained when no feasible solution exists. The approach proposed in [12] and [13] can solve problems with up to 25 data sets within reasonable time constraints and, hence, is not suitable for our goals.

We have started the implementation of a TS algorithm. In general, the TS solution is sub-optimal and quality cannot be guaranteed. However, if no feasible solution is found within a reasonable time, the TS is stopped and the negotiation process is started. Our heuristic algorithm is also more effective than a standard branch and bound technique in the search of an initial feasible solution. By interleaving heuristic techniques and negotiation, our approach can efficiently identify a feasible sub-optimum solution for query Q.

6. CONCLUSIONS AND FUTURE WORK

We have proposed a broker architecture which identifies the quasi-optimum query plan to access data from multiple syndicated data providers, with price and quality constraints.

Future work will introduce a more complex data model, in order to manage data characterized by a lower granularity, and will consider the analysis of the performance of our approach, both in terms of the quality of the heuristic solution and execution time. Furthermore, column generation techniques will be implemented in order to identify the global optimum of the problem.

ACKNOWLEDGEMENTS

This work has been partially supported by the Italian FIRB Project MAIS. Particular thanks are expressed to Prof. Carlo Batini for his invaluable suggestions.

REFERENCES

[1] Avenali, A., Bertolazzi, P., Batini, C., Missier, P. A formulation of the Data Quality Optimization Problem in Cooperative Information Systems. CAiSE Workshops (2), 2004. pp. 49–63.

[2] Ballou, D. P., Wang, R., Pazer, H.L., Tayi, G.K. Modelling Information Manufacturing Systems to Determine Information Product Quality. *Management Science*, *44*(4), 1998. pp. 462-484.

[3] Bovee, M., Srivastava, R.P., Mak, B. A conceptual framework and belief- function approach to assessing overall information quality. *Proceedings of the Sixth International Conference on Information Quality*. MIT Press, November 2001.

[4] Cappiello, C., Francalanci, C., Pernici, B. Time-related factors of data quality in multichannel information systems. *Journal of Management Information Systems*, 20(3), 2004. pp. 71–91.

[5] Eppler, M.J. *Managing Information Quality*. Springer-Verlag, 2003.

[6] Eppler, M.J., Helfert M. A Framework For The Classification Of Data Quality Costs And An Analysis Of Their Progression. *Proceedings of the Ninth International Conference on Information Quality*. MIT Press, November 2004. pp. 311-325.

[7] Faratin, P., Sierra, C., Jennings, N.R. Negotiation Decision Functions for Autonomous Agents. *Int. J. Robotics and Autonomous Systems*, 23(3-4), 1998. pp.159-182.

[8] Forino, R. Data e.Quality: Enlighten Your Users with Syndicated Data. Column published in *DMReview.com* November 19, 2001.

[9] Glover, F., Laguna, M. *Tabu Search.* Kluwer Academic Publishers, 1997.

[10] Jennings, N.R., Lomuscio, A.R.,Parsons, S., Sierra, C., and Wooldridge M. Automated Negotiation: prospects, methods, and challenges. Group Decision and Negotiation, 10(2), 2001. pp.199-215

[11] Lenzerini, M. Data integration: A theoretical perspective. *Proceedings of PODS 2002*. pp. 233–246.

[12] Leser, U., Naumann, F. Query Planning with Information Quality Bounds. *Proceedings of FQAS 2000*. pp. 85-95.

[13] Naumann, F., Leser, U., Freytag, J. C. Quality-driven Integration of Heterogeneous Information Systems. *Proceedings of VLDB 1999*. pp. 447-458.

[14] Papadimitriou, C., Steiglitz K. *Combinatorial Optimization*. Prentice Hall, 1982.

[15] Redman T.C. *Data Quality for the Information Age.* Artech House, 1996.

[16] Scannapieco M., Virgillito A., Marchetti C., Mecella M., Baldoni R. The DaQuinCIS architecture: a platform for exchanging and improving data quality in cooperative information systems. *Information Systems*, 29(7) 2004. pp. 551-582.

[17] Thompson, L. *The Mind and Heart of the Negotiator.* Prentice Hall, 2001

[18] Wang, R. Y.. Strong, D.M. Beyond Accuracy: What Data Quality Means to Data Consumers. *Journal of Management Information Systems*, 12(4) 1996. pp. 5-34.

[19] Wiederhold, G., Ceri, S. Pernici, B., Distributed database design. *IEEE Proceedings*, 1987.

[20] Wolsey, L. *Integer Programming*. Wiley, 1998.