Quality-driven Integration of Heterogeneous Information Systems

Felix Naumann
Humboldt-University of Berlin, Germany
naumann@dbis.informatik.hu-berlin.de

Ulf Leser
Technical University Berlin, Germany
leser@cs.tu-berlin.de

Johann Christoph Freytag
Humboldt-University of Berlin, Germany
freytag@dbis.informatik.hu-berlin.de

Abstract

Integrated access to information that is spread over multiple, distributed, and heterogeneous sources is an important problem in many scientific and commercial domains. While much work has been done on query processing and choosing plans under cost criteria, very little is known about the important problem of incorporating the information quality aspect into query planning.

In this paper we describe a framework for multidatabase query processing that fully includes the quality of information in many facets, such as completeness, timeliness, accuracy, etc. We seamlessly include information quality into a multidatabase query processor based on a view-rewriting mechanism. We model information quality at different levels to ultimately find a set of high-quality query-answering plans.

1 Introduction

Integrated access to information that is spread over multiple, distributed and heterogeneous sources is an important problem in many scientific and commercial domains. For instance, a current list of molecular biology information systems (MBIS) enumerates more than 400 entries [Inf98] of publicly available data sources. These can be both intensionally and extensionally overlapping, replicated, or disjoint. If we are interested in data about human genes, which has many facets, such as related diseases, genomic location, nucleotide sequence etc., we will find many potentially interesting data sources [LLRC98]. Considering them all is expensive and often infeasible.

Therefore, one of the most important tasks of data integration in such a setting is the selection of good data sources. We observed that the main user criterion for selecting sources by hand is not response time, but the expected quality of the data. Clearly, MBIS information sources store data of varying quality. Molecular biology researchers are particularly sensitive to criteria such as *timeliness*, *completeness* or *accuracy of data*. Results become outdated quickly, and the intrinsic imprecision of many experimental techniques leads to fuzzy data, where the degree of fuzziness often varies with the quality standards of the data source. But the result of the integration process is directly influenced by data quality. For example, a large company has reported, that up to 60% of the information integrated to their data warehouse was unusable due to the poor quality of the input data [Orr98].

In this work we describe a data integration system based on a global schema. The contents of data sources are described with respect to this schema in the form of assertions in a top-down fashion. The salient feature of our approach is the tight integration of classical query planning and the assessment and consideration of information quality (IQ). We extend an existing framework for query planning which is based on rules that define the semantic relationship between queries. These rules – and hence queries, not entire sources or relations – are the main targets for quality assessment. This level of granularity is necessary since many information quality criteria can neither be assigned to an entire source nor to single classes.

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the VLDB copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Very Large Data Base Endowment. To copy otherwise, or to republish, requires a fee and/or special permission from the Endowment.

Proceedings of the 25th VLDB Conference, Edinburgh, Scotland, 1999.

For instance, consider a source that stores data about genes and their location on chromosomes. This source might use two different classes to store the data, one for gene information and one for the location information. We now observe that the gene information is frequently updated, as is the location information for the X chromosome. However, data on locations of genes on the Y chromosome are treated less thoroughly. The timeliness of the data of this source can neither be described with one value for the entire source nor with one value per class. Instead, we want to assign quality measures to *queries*: a high IQ score to queries for X chromosome locations and for gene information, and a low IQ score to queries for Y chromosome locations.

Another example is a source which offers two different interfaces, for instance a simple WWW interface and a direct SQL channel. Logical query planning will consider that the SQL interface might be capable of answering more complex queries than the WWW interface - planning based on information quality will capture the update frequency, accuracy, completeness etc. of the data presented in the two interfaces.

The purpose of our system is to answer a global query by using only queries that are executable by some data source. We use a view rewrite mechanism for this purpose [Les99]. We here improve the logical planning algorithm by adding two steps: First we reduce the overall number of sources in a pre-processing phase, since certain sources are often worse than others in all criteria. We ensure not to lose any source that is unique in some aspect, i.e., the only source storing data about a certain attribute. Filtering sources is important since the planning algorithm inevitably has a time-complexity which is worst-case exponential in the number of rules and hence indirectly in the number of sources. Second, we rank all plans produced in the planning phase by evaluating query-specific and attribute-specific IQ scores following the join-structure of a plan. Eventually we execute plans with the highest information quality until a stop criterion is reached: either some best percentage of plans or until some overall quality threshold is reached.

1.1 Related work.

Despite the fact that there is much research showing the importance of information quality for businesses and users [WS96, Red98], and that many techniques have been proposed to improve and maintain quality of individual information sources [Wan98], we are not aware of any project that tries to use this quality data for structured information integration.

Database interoperability and data integration for molecular biology databases is addressed in a number of projects, such as OPM [CKM+98] or bioKleisli [DOTW97]. Usually these are loose federations in the sense of [SL90], i.e., they do not offer a global unified schema, and no project regards information quality.

Several research projects such as the GlOSS system [GGMT94] or work reported in [FKL97] focussed on the problem of source selection for *text based* information systems. However, selection is typically confined to criteria used in information retrieval systems, such as word-counting measures, or to traditional DBMS criteria such as response time. Within the DWQ project Jeusfeld et al. proposed a quality meta model to store IQ metadata [JQJ98]. However, their approach is guided by data warehouse quality requirements and a data warehouse architecture which is a special case of our mediator architecture.

Our planning method uses a local-as-view approach [Ull97] similar to the Information Manifold [LRO96]. Our notion of query correspondence assertions is an extension to query capability records as described there, in that it combines local-as-view with global-as-view modeling. In [Les98] we presented an improved algorithm for the query planning problem in this framework, which we now enhance with quality considerations.

1.2 Example.

We will use the following example throughout the paper. Our mediator is designed to provide information about genes and is modeled in its global relational schema (see Figure 1). A gene (Gn) is, very roughly speaking, a part of the human genome which is related to some property of humans (see [Rob94] for a thorough discussion on what is a gene from a computer scientist's point-of-view). Genes which are known to be related to a disease (Di) are particularly interesting. We are also interested in the sequence (Se) of a gene, which is essentially a string. Determining a sequence is a complicated and costly process. The quality of the result varies with the institution that carries out the experiments. We store this as its origin (Or). Interesting properties of a sequence, for instance the occurrence of repeats, are stored as annotation (An). Again, the quality of annotation depends highly on the effort that is invested in its analysis and differs from source to source.

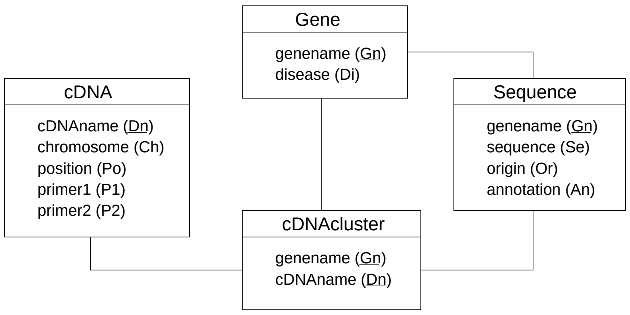

Figure 1: Information model of the mediator.

Most parts of the sequence of a human being do not contain genes. One possibility to find genes is to create pieces of so-called complementary DNA (cDNA). Locating a cDNA on a chromosome requires knowledge of two primers (P1, P2) which are short pieces of its sequence. Different cDNAs are often overlapping. They are therefore clustered into larger sections. cDNAclusters are then related to genes.

The mediator can query the five different sources which are listed below. The sources have overlapping scopes and varying information quality. For their interface relations as exposed by a wrapper see Table 1. Since this work focuses on planning using information quality, we make some simplifications, such as the use of object names as global keys.

- Source S_1 stores sequences which it copies infrequently from other sites, sometimes introducing parsing errors.
- Source S_2 also copies sequence data from other sites, also infrequently updated, but uses more sites and is hence more complete.
- Source S_3 is the WWW server of an institute which does its own sequencing. Sequence data is highly up-to-date, but few annotations are provided. The server is frequently unavailable.
- Source S_4 is a renowned commercial provider of cDNA data. It provides two interfaces: 1. A free WWW server with a slow connection. Only the chromosome location is retrievable. 2. A fast SQL connection through which clients can retrieve all attributes, but there is a charge per query. Primer sequences are available for most of their cDNAs, chromosome positions only some.
- Source S_5 is a directly accessible relational database which stores mapping and sequence data for genes. The schema (not given here) is different from our global schema. The mediator uses two queries: one relates genes and sequences, the other relates genes to their cDNAclusters.

1.3 Structure of this paper.

We describe the logical query planning in Section 2. Section 3 formally introduces information quality as a set of properties, which we classify for use in Section 4. There, we show how IQ plays a decisive role in query processing and leads to high quality results. We conclude in Section 5 and give a brief outlook to future work.

2 Logical Description of Information Sources

Our approach is based on a standard wrapper-mediator architecture (see Figure 2) with the relational model as canonical data model [Wie92]. Each information source is wrapped by one or more source-specific modules, the *wrappers*, which offer a relational export schema and query interface, hiding the particular data model, access path, and interface technology of the source. Wrappers are used by a *mediator* which offers an integrated access through its global schema. In this section we describe the logical planning of user queries. Details can be found in [Les98] and [NLF99].

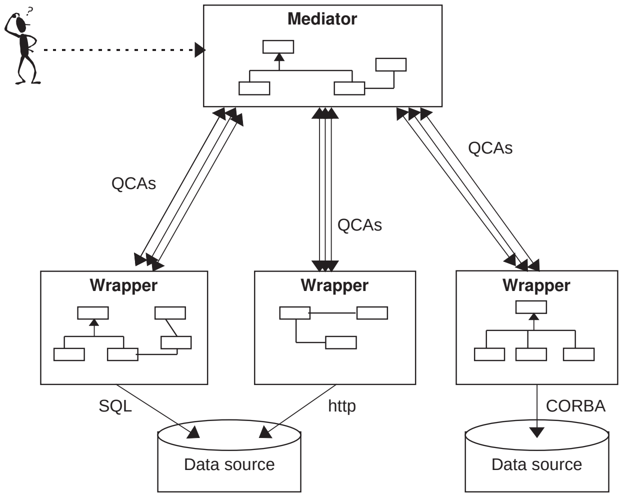

Figure 2: Principal architecture. The content of sources is described with sets of QCAs. Sources can be accessed through one or more interfaces.

To answer queries and to select sources, a mediator must know the content of each source with respect to its own schema. This *semantic knowledge* is defined by an administrator through query correspondence assertions (QCAs), which are set-oriented equations between queries against one wrapper and queries against the mediator schema. We always assume inner-join semantics. A QCA has the general form

$$MQ \leftarrow S_i.v_j \leftarrow WQ$$

where MQ (mediator query) is a conjunctive query against the mediator schema, WQ (wrapper query) is a conjunctive query against the schema of one wrapper and $S_i.v_j$ is a view which must be *safe* in both directions, i.e., variables in the view must appear in both queries. They are called *exported* variables. S_i is the source that is addressed through the QCA; internally, the mediator also bears in memory the wrapper that is used. By defining a QCA, the administrator asserts the intensional equivalence of the results of both MQ and WQ, restricted to the variables appearing in the view.

Example. The following simple QCA describes the content of S_1:

$$\begin{aligned}&\text{sequence(Gn,Se,Or,An)}\\&\quad\leftarrow S_1.v_1(Gn, Se, Or, An)\\&\quad\leftarrow \text{seq(Gn,Se,Or,An)}\end{aligned}$$

Note that this rule does not mean that S_1 is the only source storing sequence data. Others can contribute to the global relation as well.

Describing S_5 requires two QCAs, one for each of the two queries used by the mediator:

$$\begin{aligned}&\text{gene(Gn,Di), sequence(Gn,Se,-,An)}\\&\quad\leftarrow S_5.v_1(Gn, Di, Se, An)\\&\quad\leftarrow \text{genes(GID,Gn,Di), genepart(GID,PID),}\\&\qquad\text{part(PID,Se,An)}\end{aligned}$$

$$\begin{aligned}&\text{gene(Gn,-), cDNAcluster(Gn,Dn),}\\&\qquad\qquad\text{cDNA(Dn,Ch,-,P1,P2)}\\&\quad\leftarrow S_5.v_2(Gn, Dn, Ch, P1, P2)\\&\quad\leftarrow \text{clustering(Gn,Dn,CID), cluster(CID,Ch),}\\&\qquad\text{primers(Dn,P1,P2)}\end{aligned}$$

The first QCA could be used for global queries asking for gene sequences, but not for queries asking for the origin of gene sequences, since this attribute is not exported through the view. See Table 1 for the list of QCAs that describe the content of the different sources in our example. □

S_1:	QCA_1	sequence(Gn, Se, Or, An) $\leftarrow S_1.v_1(Gn, Se, Or, An)$ $\leftarrow$ seq(Gn, Se, Or, An)
S_2:	QCA_2	sequence(Gn, Se, Or, An) $\leftarrow S_2.v_1(Gn, Se, Or, An)$ $\leftarrow$ seq(Gn, Se, Or, An)
S_3:	QCA_3	sequence(Gn, Se, Or, An) $\leftarrow S_3.v_1(Gn, Se, Or, An)$ $\leftarrow$ seq(Gn, Se, Or, An)
S_4:	QCA_4	cDNA(Dn, Ch, –, –, –) $\leftarrow S_4.v_1(Dn, Ch)$ $\leftarrow$ www(Dn, Ch)
	QCA_5	cDNA(Dn, Ch, Po, P1, P2) $\leftarrow S_4.v_2(Dn, Ch, Po, P1, P2)$ $\leftarrow$ direct(Dn, Ch, Po, P1, P2)
S_5:	QCA_6	gene(Gn, Di), seq.(Gn, Se, -, An) $\leftarrow S_5.v_1(Gn, Di, Se, An)$ $\leftarrow$ genes(GID, Gn, Di), genepart(GID, PID, -), part(PID, Se, An)
	QCA_7	gene(Gn, -), cDNAcluster(Gn, Dn), cDNA(Dn, Ch, -, P1, P2) $\leftarrow S_5.v_2(Gn, Dn, Ch, P1, P2)$ $\leftarrow$ clustering(Gn, Dn, Cl, P1, P2), cluster(Cl,Ch),primers(Dn,P1,P2)

Table 1: QCAs describing the semantics of the seven possible wrapper queries.

For a given user query against the mediator schema, the mediator tries to find combinations of QCAs that are *semantically contained* [ASU79] in the user query and hence provably compute only correct results. We call such combinations *plans*. In Section 4.2 we describe our algorithm to find all correct plans. The *complete* answer to a user query with respect to the given QCAs is the union over the answers of all correct plans. For instance, the global extension of *sequence* would be the union over the extension of the three "seq"-queries in sources S_1, S_2, and S_3 (Table 1).

However, there can be prohibitively many correct plans. Consider a query asking for the sequence of a specific gene. The mediator detects that S_5 can be used for the *gene* part of the query and S_1, S_2, and S_3 for the *sequence*-part. This already sums up to three different plans. Suppose there were two more sources storing genes, then the number of correct plans would increase to nine. But if the user was, for instance, particularly interested in complete annotation, plans using S_3 are not very promising; if highly up-to-date data is required, S_1 could probably be ignored.

Note that query planning as described here is fundamentally different from classical query optimization for a relational DBMS. Our planning finds plans that are correct, but possibly generate different results, while classical optimization considers plans that all produce the same result.

3 Information Quality for Information Sources

There is no agreed definition or measure for information quality, except such general notions as "fitness for use" [TB98]. In this section we define information quality (IQ) as a set of quality criteria. Information sources and query plans achieve certain IQ scores in each criterion. We aggregate the scores to determine a total IQ score for each source and plan and then rank sources and plans accordingly. Based on this ranking we execute only the best plans over the best sources disregarding the rest.

Wang and Strong have empirically identified fifteen IQ criteria regarded by data consumers as the most important [WS96]. They classified the criteria into "intrinsic quality", "accessibility, "contextual quality", and "representational quality". Their framework has already been used effectively in industry and government. We adapt this set of criteria to our integration model and to the scope of molecular biology information systems. However, we classify the criteria in a different manner to reflect better our planning process.

3.1 IQ Classification.

It is not always sufficient to assign quality scores to entire sources. Since our planning process already uses QCAs and not entire sources as the basic level of correspondence, it is natural to assign IQ scores to QCAs. Furthermore, IQ scores can even apply at an attribute level, as a source may provide high quality information in one attribute, but lower quality in another. Thus, we distinguish three classes of quality criteria:

- **Source-specific criteria** determine the overall quality of an information source. Criteria of this category, for instance reputation, apply to all information of the source, independently of how it is obtained. IQ scores of this class stay unchanged as long as the source itself does not dramatically change.

- **QCA-specific criteria** determine quality aspects of specific queries that are computable by a source. Using this finer granularity, we can e.g. model different response times for different types of queries to the same source.

- **Attribute-specific criteria** assess the quality of an information source in terms of its ability to provide the attributes of a specific user query. The IQ scores for these criteria depend on the attributes specified in the user query, and hence, the scores for these criteria can only be determined at "query time". For instance, we described S_3 as having relatively few annotations attached to the sequences they store. In such cases, we need to define specifically that the completeness of the *annotation* attribute in QCA_3 is not very high.

3.2 IQ Criteria.

We slightly modified the set of IQ criteria by Wang and Strong in [WS96], considering the specific needs of biologists and the specific properties of existing information systems. Some criteria that are not applicable to our area of discourse or data integration model are omitted. We added the two criteria reliability and price, which play a important role for molecular biology information systems. Table 2 on the next page categorizes and summarizes all our criteria. In accordance with our integration and planning process we have found three categories: Source-specific, QCA-specific, and attribute-specific criteria. The criteria of each category are dealt with differently. A more detailed description of each criterion can be found in [NLF99]. As usual, we assume independence of the criteria.

Depending on the application domain and the structure of the available sources, the classification of criteria into the three classes may vary. For instance, if sources charge the same amount of money for each query, the price criterion should be only source-specific. If, on the other hand, a source provides data with different update frequencies, the timeliness criterion should be QCA-specific. Finally, if the information of a source can be partitioned into sets with heavily diverging IQ scores, the QCAs of this source can be split according to this partitioning. Each of the new QCAs will then receive individual IQ scores.

A problem that all projects addressing IQ are facing, is the ability to assign IQ scores in an objective manner. Some of the criteria below can not be measured but are highly subjective, such as source reputation. We suggest user profiles, i.e., sets of IQ scores for all subjective criteria that are set-up once by each user and then used for all of his or her future queries.

4 Finding the Best Sources and Plans

Creating good execution plans for a user query in any DBMS involves a search space of all plans and a cost model to compare plans with one another. Building on this analogy we define the search space in our multidatabase environment as the set of all plans that answer the user query in a semantically correct manner. However, these plans produce extensionally different results as they may involve different sources. Thus, in general more than one plan should be executed to gain a response that is as complete as possible. Due to the heterogeneity of the information sources in quality and cost we expand the traditional cost model to a quality model to valuate the plans.

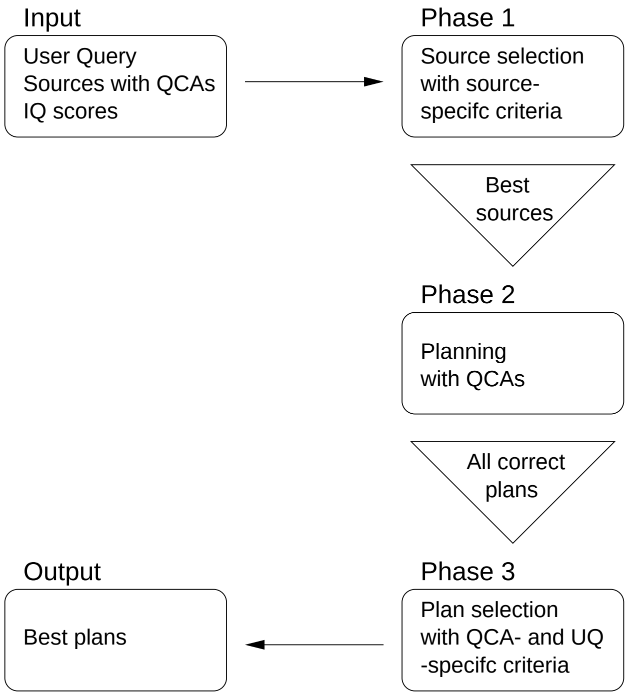

Figure 3: Three-Phase plan selection

To this end, we propose a three-phase approach to quality-driven information integration: In the first phase we reduce computational cost of the second

Class	Criterion	Brief explanation
Source-specific	Ease of understanding	User grade from 1 to 10, based on presentation of the data.
	Reputation	User grade from 1 to 10, based on personal preferences and professional experience.
	Reliability	Ranking from 1 to 10, based on accuracy of experimental method with which the data is produced.
	Timeliness	Update-frequency measured in days.
QCA-specific	Availability	Percentage of time the source is accessible, based on technical equipment and statistics.
	Price	Monetary price of a query in US Dollars. We assume a pay-by-query scheme.
	Representational Consistency	Per-query time consumption of the wrapper (for parsing, translations, etc.), in seconds. The more consistent the presentation of the source, the less work for the wrapper.
	Response Time	Average waiting time for responses, measured in seconds.
	Accuracy	Percentage of objects without data errors such as misspellings, out-of-range values, etc.
	Relevancy	Percentage of real world objects represented in the source. When the total number of real world objects is not known, an approximation can be used.
Attribute-specific	Completeness	Fullness of the relation in each attribute (horizontal fitness). Attributes typically have a certain percentage of `null`-values. Completeness of a QCA is measured as the sum over the percentages of non-`null` values in each attribute, adjusted by a user weighting stating the importance of each attribute. The weighting is specified with the user query.
	Amount	Number of attributes in the response which were not specified in the user query (vertical fitness).

Table 2: Classification of Quality Criteria for MBISs

phase by filtering out low quality sources based on the source-specific IQ criteria, continuing with only the best sources. The second phase uses the QCAs of the remaining sources to generate all correct plans, thus establishing the search space for the last phase. There we explore the entire search space using a quality model for the remaining IQ criteria and choose the best plans for execution. Figure 3 gives an overview of the three phases. For simplicity we do not apply a search strategy to combine Phases 2 and 3, rather we materialize the entire search space in Phase 2 and examine it in Phase 3.

Example. To describe each step in detail we use the example of Section 2. Table 3 gives IQ scores for each QCA and each criterion. We are aware of the difficulties of numerically expressing certain criteria, but since not the absolute IQ scores are of importance but rather their relative values, we believe that our approach is reasonable. □

To find a ranking based on multiple criteria one faces two fundamental problems: (i) The range and units of the IQ scores vary, making it necessary to scale the scores. (ii) The importance of the criteria may vary making it necessary to find a user-specific weighting of the criteria. Several *multiple attribute decision making* methods have been proposed to solve these problems [Nau98]. To find the best sources in Phase 1, we use the "Data Envelopment Analysis" method; to rank the execution plans in Phase 3 we apply the "Simple Additive Weighting" method.

4.1 Phase 1: Source Selection

Our logical planning algorithm can potentially generate an exponential number of plans in the length of the user query and the number of QCAs. Therefore, we thrive to decrease this number before we start planning. For this purpose, we use the source-specific IQ criteria to "weed out" sources that are qualitatively not as good as others. Our goal is to find a certain number or percentage of best sources independently of any user-specific weighting. The mediator performs Phase 1 only once after start-up and does not repeat it until an information source dramatically changes in a source-specific criterion, or until a new information source is added to the system.

To evaluate a large amount of sources in a *general*, user-independent way, we suggest Data Envelopment Analysis (DEA) developed by Charnes et al. as a general method to classify a population of observations [CCR78]. For a more detailed description of DEA for

	S_1		S_2		S_3		S_4			S_5		
		QCA_1		QCA_2		QCA_3		QCA_4	QCA_5		QCA_6	QCA_7
EoU(grade)	5		7		7		8			6		
Rep.(grade)	5		5		7		8			7		
Reli.(grade)	2		6		4		6			6		
Tim.(days)	30		30		2		1			7		
Av.(%)		99		99		60		80	99		95	95
Pr.(US $)		0		0		0		0	1		0	0
R.C.(sec)		1		1		.5		.7	.2		.7	.7
R.T.(sec)		.2		.2		.2		3	.1		1	1
Ac.(%)		99.9		99.9		99.8		99.95	99.95		99.95	99.95
Relev.(%)		60		80		90		80	80		60	60

Table 3: IQ scores s_{ij} of the 7 QCAs. Scores are partly inferred from the informal description in Section 1.2. Completeness and amount are not contained since they depend on the specific user query.

source selection see [NFS98]. The DEA method avoids the problems of scaling and weighting by defining an efficiency frontier as the convex hull of the unscaled and unweighted vector space of IQ dimensions. Figure 4 shows this vector space for two arbitrary IQ dimensions. Sources on the hull are defined as "good", sources below the hull as "non-good".

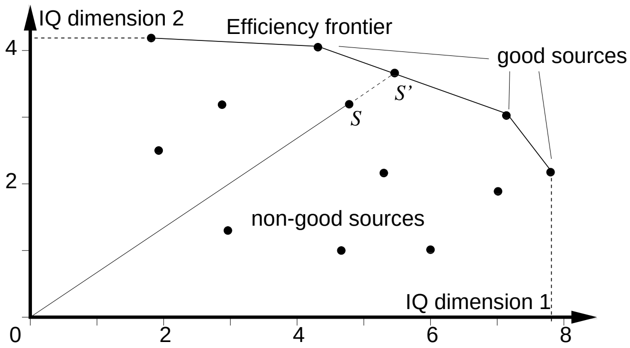

Figure 4: Classifying Sources with Data Envelopment Analysis

Consider the non-good source S in Figure 4. Assuming constant returns to scale, the virtual but realistic source S' is constructed as a convex combination of the two neighboring sources on the efficiency frontier. Clearly the virtual source S' would be better than source S, thus S is non-good.

To determine whether a source is on the frontier or below, we solve the following linear program (LP) once for each information source S_{i_0} with IQ scores s_{ij}. Please note that the variables of the LP are the weightings w_j.

$$\begin{array}{lll}
\text{maximize} & & \\
\quad IQ(S_{i_0}) := & \sum_j w_j \cdot s_{i_0 j} & \\
\text{subject to} & & \\
\quad IQ(S_i) \ = & \sum_j w_j \cdot s_{ij} & \leq 1 \ \text{ for } i = 1, \ldots, n \\
 & \qquad\qquad w_j & \geq \varepsilon > 0 \ \text{ for } j = 1, \ldots, 4
\end{array}$$

The result of each LP is the optimal quality score $IQ(S_{i_0})$ of the examined source, which is either 1 (on the frontier = good) or below 1 (below the frontier = non-good). By fine-tuning the ε-parameter we can vary the number of good sources to the desired percentage. The problem of solving such a linear program is of polynomial nature. A common way to solve LPs is the Simplex method, developed by Dantzig [Dan63], which has an exponential worst case complexity but is very efficient on average.

For further planning, we want to completely disregard non-good sources. However, there is a danger of removing a source that has low IQ but is the only source providing a certain attribute of the global schema, e.g., the only source providing chromosome data should be kept for planning, even if its IQ is low. Furthermore we must retain sources that exclusively provide certain extensions of an attribute, e.g., the only source providing data on X chromosomes should be kept, even if its IQ is low and other sources provide chromosome data for other chromosomes. Removing such sources from further consideration would "reduce" the global schema. To avoid suppressing these sources we only weed out non-good sources, whose QCAs are all contained in QCAs of good sources. In this way, the global schema remains intact.

Example. In our example only source S_1 is excluded. All other sources have an IQ score of 1 and will be further considered in the next two phases. □

4.2 Phase 2: Plan Creation

The goal of this phase is to find all combinations of QCAs that obtain semantically correct answers to a given user query. Every QCA defines a view on the global schema. We must find combinations of such views that generate correct tuples. This is equivalent to the problem of answering a query against a schema using only a set of views on the same schema. Levy et al. show that this problem is NP-complete for conjunctive queries and conjunctive view definitions [LMSS95]. In principle, one has to enumerate all combinations of views up to a certain length, and test for each of these whether it is contained in the original query. However, the worst-case exponential behavior of this algorithm only occurs in pathological cases. For instance, Chekuri and Rajaraman show that the problem is polynomial if the *width* of the query is bound [CR97].

In [Les98] we improved the algorithm by Levy et al. [LRO96], but for space limitations we here use a simpler algorithm, similar to the original one.

Example. Imagine a user query UQ asking for all genes of the X chromosome together with their related diseases, sequences, origins, and annotations. We answer this query by joining the gene relation with the sequence relation to obtain origin and annotation. We must also join the gene relation with the cDNA relation to ensure the chromosome-condition:

$$\begin{aligned} &UQ(Gn(100), Di(100), Se(100), Or(30), An(70)) \\ &\quad \leftarrow \text{gene}(Gn, Di), \text{sequence}(Gn, Se, Or, An), \\ &\quad\quad \text{cDNAcluster}(Gn, Dn), \\ &\quad\quad \text{cDNA}(Dn, Ch, \text{-}, \text{-}, \text{-}), Ch =' X'; \end{aligned}$$

□

The user weightings for each attribute are used to reflect how important they are to the user. In the example, the user expresses that he is interested in annotation and not as much in the origin of sequences. The weightings are used to compute the completeness score of plans later on.

First, for each relation of UQ we determine the set of QCAs which contain the relation in their mediator query MQ. We store this set in a *bucket* [LRO96]. We must also check if the QCAs export all necessary attributes, i.e., those that are required in UQ.

In a second step we enumerate the cartesian product of all buckets and check three conditions for each combination: (1) if it is satisfiable, (2) if it is semantically contained in UQ, and (3) whether it can be minimized, i.e., whether certain QCAs are redundant.

Example. For the four relations of UQ we construct the buckets

$$\begin{array}{lcl} \text{bucket(gene)} & = & \{QCA_6\} \\ \text{bucket(sequence)} & = & \{QCA_2, QCA_3\} \\ \text{bucket(cDNAcluster)} & = & \{QCA_7\} \\ \text{bucket(cDNA)} & = & \{QCA_4, QCA_5, QCA_7\} \end{array}$$

QCA_7 does not occur in bucket(gene) because it does not export the required Di attribute; the same holds for QCA_6 in bucket(sequence) and attribute Or. QCA_1 does not appear in bucket(sequence) since S_1 was deleted from the set of sources in Phase 1.

After enumerating the cartesian product of all buckets and checking the three conditions, we end up with the following plans, each possibly producing a different set of correct tuples for UQ.

$$\begin{aligned} P_1 &= QCA_6 \bowtie QCA_2 \bowtie QCA_7 \bowtie QCA_4 \\ P_2 &= QCA_6 \bowtie QCA_2 \bowtie QCA_7 \bowtie QCA_5 \\ P_3 &= QCA_6 \bowtie QCA_2 \bowtie QCA_7 \\ P_4 &= QCA_6 \bowtie QCA_3 \bowtie QCA_7 \bowtie QCA_4 \\ P_5 &= QCA_6 \bowtie QCA_3 \bowtie QCA_7 \bowtie QCA_5 \\ P_6 &= QCA_6 \bowtie QCA_3 \bowtie QCA_7 \end{aligned}$$

□

A plan is executed by computing the wrapper queries of the QCAs, propagating variables bindings from QCA to QCA as usual. If a WQ is executed, the resulting tuples are temporarily stored in an instance of the mediator schema. Missing values are padded with null or relationship-preserving key values that are generated automatically. The original query UQ is computed on this instance after all results are retrieved.

The mediator also keeps track of the origin of each value. Therefore, for each tuple that is obtained from a source, the mediator stores the name of this source together with the interface that was used in an extra attribute. This information is presented to the user together with the final result, which allows to further judge the information quality based on personal preferences.

4.3 Phase 3: Plan Selection

The goal of this phase is to qualitatively rank the plans of the previous phase and ultimately to restrict plan execution to some best percentage of plans, or alternatively, to as many plans as necessary to meet certain cost- or quality-constraints.

Following the DBMS approach of cost models for query execution plans with a tree-structure, we define a quality model for the tree-structured plans created in Phase 2. Leaves represent QCAs which deliver the base data. Those data are subsequently processed within the inner nodes of the tree, which represent inner join operators performed by the mediator.

Plan selection proceeds in three steps: First the IQ scores of the QCAs are determined (3a). Then the quality model aggregates these scores along tree paths to gain an overall quality score at the root of the tree, which forms the score of the entire plan (3b). Finally, this score is used to rank all plans (3c).

4.3.1 Phase 3.a: QCA Quality.

An IQ vector of length 8 is attached to each QCA, i.e., one dimension for each non-source-specific criterion. QCA-specific criteria have fixed scores for each QCA. They are determined only once or whenever the corresponding source undergoes major changes. The scores of the attribute-specific criteria completeness and amount on the other hand are recalculated for each user-query. The general IQ vector for QCAs is (abbreviated):

$$\begin{aligned}IQ(QCA_i) &:= (s_{i5}, \ldots, s_{i11})\\ &= (Av, Pr, RC, RT, Ac, Rel, Com(UQ), Am(UQ))\end{aligned}$$

Example. We determine the following IQ vectors for the QCAs participating in plans P_1 through P_6. The first six elements of each vector are taken from Table 3, the remaining two elements completeness and amount are calculated using the attribute set and attribute weighting of the user query UQ of Section 4.2.

$$\begin{aligned}IQ(QCA_2) &= (99, 0, 1, .2, 99.9, 80, 52.8, 0)\\ IQ(QCA_3) &= (60, 0, .5, .2, 99.8, 90, 49, 0)\\ IQ(QCA_4) &= (80, 0, .7, 3, 99.95, 80, 20, 1)\\ IQ(QCA_5) &= (99, 1, .2, .1, 99.95, 80, 20, 4)\\ IQ(QCA_6) &= (95, 0, .7, 1, 99.95, 60, 48.2, 0)\\ IQ(QCA_7) &= (95, 0, .7, 1, 99.95, 60, 38, 3)\end{aligned}$$

□

Up to this point, each leaf node of each plan-tree is assigned an IQ vector. However, we have no total IQ vectors for the plans yet. These scores are obtained through the quality model in the next phase.

4.3.2 Phase 3.b: Plan Quality.

Corresponding to the idea of *cost models* for DBMSs, we have designed a *quality model* to calculate the total IQ score of a plan. Since we only consider join-operators, a plan is a binary tree with QCAs as leaves and join operators as inner nodes. The IQ vector for an inner join node is calculated as a combination of the IQ vectors of its left and right child nodes l and r, as shown in Equation (1).

$$\begin{aligned}IQ(l \bowtie r) &:= IQ(l) \circ IQ(r)\\ &= (s_{l5} \circ s_{r5}, \ldots, s_{l12} \circ s_{r12}) \qquad (1)\end{aligned}$$

Figure 5 shows the plan tree for P_3 with its aggregated IQ vectors. In each criterion the IQ scores s_{ij} are computed with the $\circ$-operator (or "merge function") which is resolved according to Table 4. Since all merge functions in Table 4 are both commutative and associative, a change of the join execution order within a plan has no effect on its IQ score. This is desirable, since the user perceives the quality of the query result and not the quality of how this result is obtained. Furthermore, we do not consider the execution time of joins performed by the mediator since we assume that execution time is dominated by the response times of the sources.

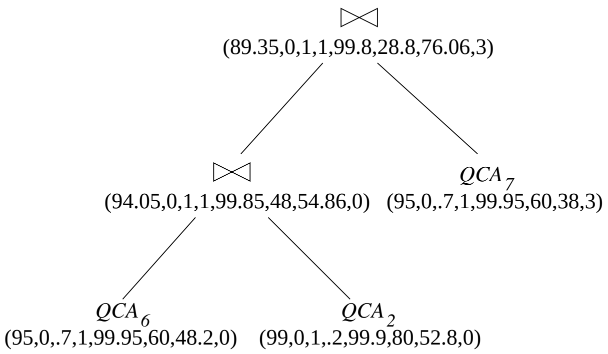

Figure 5: Merging IQ vectors in join nodes in plan P_3

Example. The six plans have aggregated IQ vectors

$$\begin{aligned}IQ(P_1) &= (71.00, 0, 1, 3, 99.75, 23.04, 78.06, 4)\\ IQ(P_2) &= (88.45, 1, 1, 1, 99.75, 23.04, 78.06, 7)\\ IQ(P_3) &= (89.35, 0, 1, 1, 99.80, 28.80, 76.06, 3)\\ IQ(P_4) &= (43.32, 0, .7, 3, 99.65, 25.92, 75.94, 4)\\ IQ(P_5) &= (53.61, 1, .7, 1, 99.65, 25.92, 75.94, 7)\\ IQ(P_6) &= (54.50, 0, .7, 1, 99.70, 32.40, 73.94, 3)\end{aligned}$$

□

Up to this point, the scores are neither scaled nor weighted, making a comparison or ranking of plans impossible.

4.3.3 Phase 3.c: Plan Ranking.

The IQ scores of the vectors must be scaled, weighted, and compared to find a total IQ score for each plan and thus a ranking of the plans. To this end, we use the Simple Additive Weighting (SAW) method. It is one of the simplest but well perceived decision making methods, in that its ranking results are usually very close to results of more sophisticated methods [Nau98]. The method is comprised of three basic steps: scale the scores to make them comparable, apply the user weighting, and sum up the scores for each source.

The IQ scores of the criteria availability, accuracy, relevancy, and completeness are scaled according to Equation (2) below, where $s_j^{\min}$ and $s_j^{\max}$ are the minimum and maximum score in criterion j respectively. The criteria price, representational consistency, response time, and amount are negative criteria, i.e., the higher the score, the worse the quality. Thus, they are scaled according to Equation (3). With these scaling functions all scores are in $[0, 1]$, the best score of any criterion obtains the value 1, and the worst score of any criterion obtains the value 0. This property assures comparability of scores across different criteria and in different ranges.

$$v_{ij} := \frac{s_{ij} - s_j^{\min}}{s_j^{\max} - s_j^{\min}} \qquad (2)$$

$$v_{ij} := \frac{s_j^{\max} - s_{ij}}{s_j^{\max} - s_j^{\min}} \qquad (3)$$

Criterion	Merge function "$\circ$"	Brief explanation
Availability	$s_{l5} \cdot s_{r5}$	Probability that both sites are accessible.
Price	$s_{l6} + s_{r6}$	Both queries must be payed.
Repr. Consistency	$\max[s_{l7}, s_{r7}]$	Wrapper integrates sources in parallel.
Response Time	$\max[s_{l8}, s_{r8}]$	Both children are processed in parallel.
Accuracy	$s_{l9} \cdot s_{r9}$	Probability that left and right side do not contain an error.
Relevancy	$s_{l10} \cdot s_{r10}$	Probability for join match.
Completeness	$s_{l11} + s_{r11} - s_{l11} \cdot s_{r11}$	Probability that either left or right side has non-`null` value (operations at attribute level).
Amount	$s_{l12} + s_{r12}$	All unnecessary attributes must be dealt with.

Table 4: Merge functions for Quality Criteria

For the weighting step SAW requires a weight-vector $W = (w_1, \ldots, w_m)$ specified by the user such that $\sum_{j=1}^{m} w_j$ equals 1. The weight-vector reflects the importance of the individual criteria to the user. We store the user-specific weight-vectors in a profile. Hence, the inclusion of quality reasoning is completely transparent to a user once the system is set up, i.e., once all QCAs and sources have their IQ scores. The only exception is the possibility to weight specific attributes of a query. However, we believe that most users would appreciate this as a benefit rather than a burden.

For a plan P_i the overall quality score $IQ(P_i)$ is calculated as the weighted sum according to Equation (4):

$$IQ(P_i) := \sum_{j=1}^{m} w_j \cdot v_{ij} \tag{4}$$

The final IQ score of the plan again is in $[0, 1]$ and gives the ranking position of the plan. After the IQ scores for all plans have been calculated, we choose and execute a certain number of best plans.

Example. With the indifferent weighting vector $W = (\frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8})$ where all criteria have equal importance, the following IQ scores are obtained (in ranking order):

$$IQ(P_3) = .7663 \tag{1}$$
$$IQ(P_6) = .697 \tag{2}$$
$$IQ(P_1) = .5023 \tag{3}$$
$$IQ(P_2) = .4559 \tag{4}$$
$$IQ(P_4) = .4429 \tag{5}$$
$$IQ(P_5) = .3771 \tag{6}$$

A user preferring quick response time at any price, might specify the weighting $W = (\frac{1}{8}, \frac{0}{8}, \frac{1}{8}, \frac{2}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8})$ obtaining a ranking of the plans in the order P_3, P_6, P_2, P_5, P_1, P_4. Plan P_1 is ranked lower than before because it includes QCA_4 which has a very high response time. Despite its high price, Plan P_2 is ranked higher than before since it has a low response time. □

With the exception of the completeness criterion, all merge functions decrease the aggregated IQ scores with each additional QCA. Thus, there is a natural tendency favoring short plans, i.e., plans consisting of few QCAs. Not only does this reflect the influence of the criteria, it also conforms to intuition: Biologists will probably not be happy to accept results where the four attributes of the query are generated in four different sources.

The complexity of Phase 3 is linear in the number of considered plans times the maximum length of the plans. This length is in turn bounded by the number of relations in the user query.

5 Conclusion and Outlook

We have proposed a novel method to the well known and important, yet frequently ignored problem of considering information quality in information integration. This problem has not, to our best knowledge, been adequately addressed before. Our results offer a solution to the notorious problem of information overload, based on a filtering of important information with the help of a rich set of quality criteria. Using these criteria quality-driven information integration identifies high quality plans which produce high quality results.

Clearly, the selection of quality criteria is a subjective task. Yet our method is by no way restricted to the criteria we used in this paper. We have described merge functions for each criterion which calculate the quality of the information in a join result. Due to the associativity of these merge-functions, we determine the quality of a result independently of how this result is created. This freedom will allow us to include binding patterns in the QCAs, which often dictate a specific join order. Furthermore, a traditional post-optimization can be performed to find the best join order of the chosen plans without influencing their quality score.

Future work will also include a tighter cooperation of the plan creation phase and plan selection phase: Information quality scores can be used in a branch & bound fashion to dramatically improve planning time. Lifting our current model to a higher level, we plan not only to calculate the quality of plan results, but also the quality of the union of several plans to find the best combination of plans to execute. We believe that the same principles of calculating and merging quality scores apply.

Acknowledgements

This research was supported by the German Research Society, Berlin-Brandenburg Graduate School in Distributed Information Systems (DFG grant no. GRK 316). We also thank Myra Spiliopoulou for many helpful comments.

References

[ASU79] Alfred V. Aho, Yehoshua Sagiv, and Jeffrey D. Ullman. Efficient optimisation of a class of relational expressions. *ACM Transactions on Database Systems*, 4(4):435–454, 1979.

[CCR78] A. Charnes, W.W. Cooper, and E. Rhodes. Measuring the efficiency of decision making units. *European Journal of Operational Research*, 2:429–444, 1978.

[CKM+98] I-Min A. Chen, Anthony S. Kosky, Victor M. Markowitz, Christoph Sensen, Ernest Szeto, and Thodoros Topaloglou. Advanced query mechanisms for biological databases. In *6th Int. Conf. on Intelligent Systems for Molecular Biology*, pages 43–51, Montreal, Canada, 1998. AAAI Press, Menlo Park.

[CR97] Chandra Chekuri and Anand Rajaraman. Conjunctive query containment revisited. In *6th Int. Conference on Database Theory; LNCS 1186*, pages 56–70, Delphi, Greece, 1997.

[Dan63] G.B. Dantzig. *Linear Programming and Extensions.* Princeton University Press, Princeton, NJ, 1963.

[DOTW97] Susan Davidson, G. Christian Overton, Val Tannen, and Limsoon Wong. BioKleisli: A digital library for biomedical researchers. *Int. Journal on Digital Libraries*, 1:36–53, 1997.

[FKL97] Daniela Florescu, Daphne Koller, and Alon Levy. Using probabilistic information in data integration. In *Proc. of the 23rd VLDB Conference*, Athens, Greece, 1997.

[GGMT94] Luis Gravano, Hector Garcia-Molina, and Anthony Tomasic. The effectiveness of GlOSS for the text database recovery problem. In *Proc. of the ACM SIGMOD Conference*, 1994.

[Inf98] InfoBiogen. DBCat - the public catalog of databases. WWW Page http://www.infobiogen.fr/services/dbcat, Infobiogen, France, 1998.

[JQJ98] M.A. Jeusfeld, C. Quix, and M. Jarke. Design and analysis of quality information for data warehouses. In *Proc. of the 17th Int. Conf. on Conceptual Modeling (ER98)*, Singapore, November 1998.

[Les98] Ulf Leser. Combining heterogeneous data sources through query correspondence assertions. In *Workshop on Web Information and Data Management*, Washington, D.C., 1998.

[Les99] Ulf Leser. Designing a global information resource for molecular biology. In *8th GI Fachtagung: Datenbanksysteme in Buero, Technik und Wissenschaft*, Freiburg, Germany, 1999. to appear.

[LLRC98] Ulf Leser, Hans Lehrach, and Hugues Roest Crollius. Issues in developing integrated genomic databases and application to the human X chromosome. *Bioinformatics*, 14(7):583–690, 1998.

[LMSS95] Alon Y. Levy, Alberto O. Mendelzon, Yehoshua Sagiv, and Divesh Srivastava. Answering queries using views. In *14th ACM PODS*, pages 95–104, San Jose, CA, 1995.

[LRO96] Alon Y. Levy, Anand Rajaraman, and Joann J. Ordille. Querying heterogeneous information sources using source descriptions. In *22th VLDB*, pages 251–262, Bombay, India, 1996.

[Nau98] Felix Naumann. Data fusion and data quality. In *Proc. of the New Techniques & Technologies for Statistics Seminar (NTTS)*, Sorrento, Italy, 1998.

[NFS98] Felix Naumann, Johann Christoph Freytag, and Myra Spiliopoulou. Quality-driven source selection using Data Envelopment Analysis. In *Proc. of the 3rd Conference on Information Quality (IQ)*, Cambridge, MA, 1998.

[NLF99] Felix Naumann, Ulf Leser, and Johann Christoph Freytag. Quality-driven integration of heterogenous information systems. Technical Report 117, Humboldt-Universität zu Berlin, 1999.

[Orr98] Ken Orr. Data quality and systems theory. *Communications of the ACM*, February 1998.

[Red98] Thomas C. Redman. The impact of poor data quality in the typical enterprise. *Communications of the ACM*, 41(2):79–82, 1998.

[Rob94] Robert J. Robbins. Representing genomic maps in a relational database. In Sandor Suhai, editor, *Computational Methods in Genome Research*, pages 85–96. Plenum Press, New York, 1994.

[SL90] Amit Sheth and James A. Larson. Federated database systems for managing distributed, heterogeneous and autonomous databases. *ACM Computing Survey*, 22(3):183–236, 1990.

[TB98] Giri Kumar Tayi and Donald P. Ballou. Examining data quality. *Communications of the ACM*, 41(2):54–57, 1998.

[Ull97] Jeffrey D. Ullman. Information integration using logical views. In *6th ICDT*, pages 19–40, Delphi, Greece, 1997.

[Wan98] Richard Y. Wang. A product perspective on Total Data Quality Management. *Communications of the ACM*, 41(2):58–65, 1998.

[Wie92] Gio Wiederhold. Mediators in the architecture of future information systems. *IEEE Computer*, 25(3):38–49, 1992.

[WS96] Richard Y. Wang and Diane M. Strong. Beyond accuracy: What data quality means to data consumers. *Journal on Management of Information Systems*, 12, 4:5–34, 1996.