Available at
www.ComputerScienceWeb.com
POWERED BY SCIENCE DIRECT®

Data & Knowledge Engineering 47 (2003) 105–129

DATA & KNOWLEDGE ENGINEERING

www.elsevier.com/locate/datak

Estimating the quality of answers when querying over description logic ontologies

Martin Peim [a,*], Enrico Franconi [b], Norman W. Paton [a]

[a] *Department of Computer Science, University of Manchester, Oxford Road, Manchester M13 9PL, UK*
[b] *Free University of Bolzano, Piazza Domenicani 3, 39100 Bolzano, Italy*

Received 5 November 2002; received in revised form 5 February 2003; accepted 26 March 2003

Abstract

Information integration systems allow users to express queries over high-level conceptual models. However, such queries must subsequently be evaluated over collections of sources, some of which are likely to be expensive to use or subject to periods of unavailability. As such, it would be useful if information integration systems were able to provide users with estimates of the consequences of omitting certain sources from query execution plans. Such omissions can affect both the *soundness* (the fraction of returned answers which are returned) and the *completeness* (the fraction of correct answers which are returned) of the answer set returned by a plan. Many recent information integration systems have used conceptual models expressed in description logics (DLs). This paper presents an approach to estimating the soundness and completeness of queries expressed in the $\mathcal{ALCQI}$ DL. Our estimation techniques are based on estimating the cardinalities of query answers. We have have conducted some statistical evaluation of our techniques, the results of which are presented here. We also offer some suggestions as to how estimates for cardinalities of subqueries can be used to aid users in improving the soundness and completeness of query plans.

Keywords: Data quality; Cardinality estimation; Description logics; Distributed query processing; Information integration

* Corresponding author.
E-mail addresses: peim@cs.man.ac.uk (M. Peim), franconi@inf.unibz.it (E. Franconi), paton@cs.man.ac.uk (N.W. Paton).

0169-023X/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/S0169-023X(03)00067-3

1. Introduction

In recent years, a number of distributed query-processing systems have been developed in which the global schema and user queries are expressed in some form of description logic (DL) (for example, TAMBIS [1], DWQ [2], Information Manifold [3], PICSEL [4], SIMS [5]). The use of a DL as both high-level data description language and query language has several advantages:

- it provides an expressive global modelling language;
- it enables the use of reasoning support for the DL to assist in building consistent conceptual models [2];
- experience [6,7] has shown that reasoning support is useful in assisting users in the formation of queries which accurately reflect their needs; and
- intensional knowledge captured in a DL ontology can be used in query translation and optimisation [9];

Typically in these systems, the first stage of query processing is to rewrite user queries expressed over some DL ontology into DL expressions using only atoms that can be mapped to source databases where each source contains a set of instances of some atomic DL concept or role. This mapping of sources to concepts and query rewriting may be done using either a *global-as-view* (e.g., SIMS, TAMBIS) or a *local-as-view* approach (e.g., DWQ, Information Manifold, PICSEL). A comparison between the two approaches is given in [8]. Since our methods are applied to the rewritten query, they could be applied (with suitable modifications to adapt them to different DLs) to either type of system.

For some concepts and roles there may be several sources available to provide extents. The user of the system may wish to use only some of the available sources to reduce financial costs or response times. Furthermore, some sources may be unavailable at any given time, and the user may need to decide whether to wait for all sources to be available or to proceed with the query using the sources that are currently available. In order to evaluate such trade-offs, we need to be able to estimate the amount of information lost by only using a subset of the existing data, and the effects of such information loss on the quality of query answers. It is this problem that is addressed in this paper.

If the DL query language contains non-monotonic operations such as negation and universal quantification (under a closed-world assumption), the omission of source data can lead to the inclusion of incorrect answers in the query answer set, as well as the exclusion of correct answers. One therefore needs to be concerned about both *incompleteness* (''how many of the answers to the query do we fail to retrieve?'') and *unsoundness* (''how many of the answers we retrieve are incorrect?'') of query plans.

In [9], we describe a (global-as-view) distributed DL query-processing system (with multiple sources for each source atom) in which queries are formulated in a relatively expressive DL, $\mathcal{ALCQI}$. This paper presents a method for estimating soundness and completeness of $\mathcal{ALCQI}$ query plans, by estimating the cardinality of the extents of associated $\mathcal{ALCQI}$ expressions (see [10] for an account of the estimation the cardinalities of intermediate results in relational algebra expressions in the context of database query optimisation). Our methods make use of statistical

information which can be gathered from the source databases in advance of query processing (for comparison, in [11], the authors use information from database profiles and statistical sampling to estimate cardinalities of derived relations in database query processing).

Soundness and completeness of source data and query answers can be thought of as measures of data quality (see [12] for a classification of data quality dimensions). In recent years, much attention has been given to data quality issues in information systems research. A tool for data quality analysis using data profiling is described in [13]. Data quality issues in the setting of Data Warehousing are addressed in [14]. In particular an approach is presented to the problem of satisfying data currency constraints when answering queries.

An extension of Wang's Total Data Quality Management [15] (TDQM) framework to the domain of Cooperative Information Systems (CIS) is proposed in [16]. Cooperating organisations are expected to export conceptual models of the quality of their data along with models of the data, a data quality model is proposed for expressing data quality schemas and an indication is given of how an extended TDQM cycle might be used to manage data quality in a cooperative environment. A proposal for a service-based framework, using XML for underlying data and quality models, and a data quality broker service for the management of data quality issues in a CIS is given in [17]. The broker service can supply query answers meeting users' quality specifications, and can also offer potentially more reliable data from previous query answers to information providers. In contrast to these approaches, our work is focused on assessing the quality of query *answers*, rather than improving the quality of the source data.

In [18], the authors propose a model for specifying soundness and completeness of relational database table instances, and a method of using these specifications to estimate the quality of query answers, working in an extension of relational algebra. They also show how their techniques can be used to assist in value-conflict resolution in the Multiplex multidatabase system. Our work is concerned with similar problems in the case where the the high-level schema language is a DL. We do not require the sources to publish additional metadata but can rely on cardinality statistics which can be gathered by examining the source data. Our emphasis also differs from [18] in that we consider soundness or completeness of one query with respect to another, rather than of a query answer with respect to a notional real-world answer.

The notion of completeness as a quality measure in distributed query planning is also considered (along with other quality criteria such as response time, accuracy and timeliness) in [19] (our notion of *completeness* corresponds to their notion of *relevancy*). A focus in [19] is on exploiting quality information in query planning, which is further pursued in [20]. However, the query language considered in these papers is much simpler than a DL. For example, it has no negation, so that soundness is not an issue, and the only merge operation in query planning is *join*.

In [21], an approach is presented to the selection and ranking of information sources and unions of sources based on the grounds of content and quality metadata provided by those sources. The application domain considered is scientific data on the World Wide Web. Completeness is one of the quality dimensions considered, although in contrast to our work and that in [19], sources are only combined by *union* operations with no *join*s being performed.

Information integration systems typically provided users with a high-level query interface which is designed to hide the details of individual informations sources. When quality issues are of concern, it is useful to relax this source transparency somewhat to allow users to evaluate data

quality and make selections between sources. Section 7 of this paper indicates how this might be done in our framework. A more fine-grained approach to removal of source transparency can be found in [22], which describes a data lineage facility associated to a mediator-based data integration system, in which the provenance of each query result can be traced.

Closely related to data quality is the area of data cleansing, which is concerned with improving data by removing errors and inconsistencies. An overview of data cleansing is given in [23], which also gives a comparison of several methods of detecting errors in data sets. In [24], a data cleaning framework is proposed which incorporates an exception mechanism and a data lineage facility. The lineage facility enables the tuning of data cleaning programs. A scheme is proposed in [25] for handling data quality issues arising from data integration by providing data quality metadata (in the form of possibly time-dependent quality comparison assertions) which can be used to modify dynamically the resolution of semantic conflicts between sources.

The paper is organised as follows. Section 2 contains a brief account of query processing over DLs, in particular $\mathcal{ALCQI}$. Section 3 presents our definitions of soundness and completeness for query plans. Section 4 describes our estimation method for concept cardinalities. Section 5 presents some results of statistical testing of our methods. Our estimation methods rely on the availability of statistics for the cardinalities of conjunctions of atomic concepts. Since values for all these cardinalities are unlikely to be available in practice, we present a simple estimation technique in Section 6. Section 7 is concerned with how the user interface may be used to help users to improve the quality of their query plans by suggesting which subqueries would benefit most from consulting additional data sources. Finally, Section 8 contains some conclusions and suggestions for further work.

2. Query answering over the description logic $\mathcal{ALCQI}$

The basic types of a DL are *concepts* and *roles*. A concept expression is a description gathering the common properties among a collection of individuals; from a logical point of view it is a unary predicate ranging over the domain of individuals. Inter-relationships between these individuals are represented by means of role expressions (which are interpreted as binary relations over the domain of individuals). Roles can be seen as denoting properties of individuals, since they associate values for the property to individuals of a given class.

$\mathcal{ALCQI}$ is a DL featuring a rich combination of constructors, including full boolean operators, qualified number restrictions, inverse roles and general inclusion assertions. The syntax rules at the left-hand side of Fig. 1 define valid concept and role expressions. In the figure, C and R denote concept and role expressions, A is an atomic concept name and P is an atomic role name.

The semantics of $\mathcal{ALCQI}$ can be given in terms of an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consisting of a non-empty set $\Delta^{\mathcal{I}}$ of individuals (the *domain* of $\mathcal{I}$) and a function $\cdot^{\mathcal{I}}$ (the *interpretation function* of $\mathcal{I}$) satisfying the rules given on the right-hand side of Fig. 1. Every concept is interpreted as a subset of $\Delta^{\mathcal{I}}$ and every role as a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. An *ontology* (or *knowledge base*) is a finite set Σ of axioms of the form $C \sqsubseteq D$, involving concept or role expressions C, D; we write $C \doteq D$ as a shortcut for both $C \sqsubseteq D$ and $D \sqsubseteq C$. An interpretation $\mathcal{I}$ *satisfies* $C \sqsubseteq D$ if and only if the interpretation of C is included in the interpretation of D, i.e., $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. The set of admissible interpretations for an ontology is limited to those which satisfy the axioms. Given concept or role

$$
\begin{array}{rll}
C, D \rightarrow & A \mid & \\
& \top \mid & \top^{\mathcal{I}} = \Delta^{\mathcal{I}} \\
& \bot \mid & \bot^{\mathcal{I}} = \emptyset \\
& \neg C \mid & (\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} \\
& C \sqcap D \mid & (C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}} \\
& C \sqcup D \mid & (C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}} \\
& \forall R.C \mid & (\forall R.C)^{\mathcal{I}} = \{i \in \Delta^{\mathcal{I}} \mid \forall j.\, R^{\mathcal{I}}(i,j) \Rightarrow C^{\mathcal{I}}(j)\} \\
& \exists R.C \mid & (\exists R.C)^{\mathcal{I}} = \{i \in \Delta^{\mathcal{I}} \mid \exists j.\, R^{\mathcal{I}}(i,j) \wedge C^{\mathcal{I}}(j)\} \\
& \exists^{\geq n} R.C \mid & (\exists^{\geq n} R.C)^{\mathcal{I}} = \{i \in \Delta^{\mathcal{I}} \mid \sharp\{j \in \Delta^{\mathcal{I}} \mid R^{\mathcal{I}}(i,j) \wedge C^{\mathcal{I}}(j)\} \geq n\} \\
& \exists^{\leq n} R.C & (\exists^{\leq n} R.C)^{\mathcal{I}} = \{i \in \Delta^{\mathcal{I}} \mid \sharp\{j \in \Delta^{\mathcal{I}} \mid R^{\mathcal{I}}(i,j) \wedge C^{\mathcal{I}}(j)\} \leq n\} \\
R \rightarrow & P \mid & \\
& R^{-} & (R^{-})^{\mathcal{I}} = \{(i,j) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid R^{\mathcal{I}}(j,i)\}
\end{array}
$$

Fig. 1. Syntax of $\mathcal{ALCQI}$ concept expressions (C, D) and role expressions (R), and their semantics under an interpretation $\mathcal{I}$. A is an atomic concept name and P is an atomic role name. There are no restrictions on the interpretations of atomic names.

expressions C and D, if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for all admissible interpretations $\mathcal{I}$, we say that C is *subsumed* (or *contained*) by D. Since our estimation methods do not take axioms into account, we omit any further discussion of them. Given an ontology, a *legal database* is a finite admissible interpretation for it.

As an example, we consider an application dealing information about students at a particular university. The ontology for this application refers, among others, to the following atomic concepts:

- science-std, whose instances are all students registered in the Science Faculty.
- s-std, representing students in the Computer Science Department.
- phys-std, representing students registered with the Physics Department.
- hall-std, representing students living in university accommodation.

There is also a role flat-mate, whose instances are pairs of students who share their accommodation (for the purpose of this example the flat-mate relationship is assumed to be irreflexive—one is not considered to be one's own flat-mate). The ontology contains axioms such as

- cs-std $\sqsubseteq$ science-std, which declares that all CS students are science students.
- cs-std $\sqcap$ phys-std $\sqsubseteq \bot$, which states that no student can be registered in both CS and Physics.
- hall-std $\sqsubseteq \forall$flat-mate.hall-std, which expresses the constraint that, if a student lives in a hall, then so do all of his or her flat-mates.

A *query* over an $\mathcal{ALCQI}$ ontology is simply a concept expression. Some examples of queries over our example ontology are:

- $Q_1 = \mathsf{hall\text{-}std} \sqcup \mathsf{science\text{-}std}$, which asks for all students who either are registered in the Science Faculty or live in university accommodation.
- $Q_2 = \mathsf{hall\text{-}std} \sqcap \neg\mathsf{science\text{-}std}$ which asks for all non-science students living in halls.
- $Q_3 = \mathsf{hall\text{-}std} \sqcap \exists^{\leqslant 1}\mathsf{flat\text{-}mate.science\text{-}std}$ which asks for all hall students, at most one of whose flat-mates is registered with the Science Faculty.

Given a legal database, the *answer* to a query is simply the interpretation of the query concept expression with respect to the legal database, i.e., it is the set of *instances* of the query concept expression itself. We restrict our attention to *safe* queries, where a query is considered safe if answering that query does not involve looking up information not referred to in the query. This is crucial to restrict the scope of a query. For example, the query $\forall R.C$ is unsafe because answering it involves, among other things, finding all individuals with no R-fillers, and this information cannot be obtained by examining the instances of R and C. In the presence of incomplete information, we cannot reasonably expect to estimate cardinalities of unsafe queries, since that would involve estimating the cardinality of "the rest of the world". To check if a query Q is safe we rewrite it into a negation normal form [1] Q'. Then Q is safe if it has the form $\bot$, A (where A is atomic), $\exists R.C$ or $\exists^{\geqslant n}R.C$ ($n \geqslant 1$). It is unsafe if it has the form $\top$, $\neg A$, $\forall R.C$ or $\exists^{\leqslant n}R.C$. A conjunction is safe if and only if at least one of its conjuncts is safe. A disjunction is safe if and only if all of its disjuncts are safe. Note that, under this definition, a concept expression is safe if and only if its negation is unsafe.

We present below a proposal for estimating soundness and completeness of query plans. The measures we use are based on estimations of the cardinalities of concept extensions. We assume throughout that we have a fixed $\mathcal{ALCQI}$ ontology B and a fixed (finite) interpretation $\mathcal{I}$ of B. For any concept expression C over the symbols of B we define the *cardinality* of C, denoted $\#(C)$, to be the number of elements in its extension $C^{\mathcal{I}}$.

3. Soundness and completeness

In order to assess trade-offs between soundness and completeness of query plans and factors such as cost and availability of data sources, we need quantitative measures of soundness and completeness. In this section we define a pair of such measures. They are based on the cardinalities of DL concepts. We will express our definitions in terms of the $\mathcal{ALCQI}$ DL, but similar definitions could be made for other DLs.

We assume that all queries are expressed in terms of a set of basic concept and role names (atoms) and that for each basic atom we have a set of sources available. In a complete query-processing system, we would have other concept names in the ontology and a mechanism for rewriting user queries into queries involving only the basic names. Each source for a concept C is represented by a concept name C' which is *subsumed* by C—this is called the *sound view assumption*. That is, in any admissible model for the ontology, each instance of the C' is an instance

[1] By pushing negations inwards in the usual way, one can rewrite any $\mathcal{ALCQI}$ concept expression into an equivalent expression in *negation normal form* or NNF, where negations only appear in front of concept names.

of C. Similarly, each role source for a role R is a role R' which is subsumed by R. The representing concepts and roles will be referred to as *source concepts* and *source roles*.

In our example application, we assume that hall-std, cs-std and phys-std are source concepts, whose instances are to be retrieved from databases maintained by the University Accommodation Office, the CS Department and the Physics Department, respectively. The role flat-mate is a source role, also with a single source. The sources for the concept science-std are cs-std and phys-std (perhaps this is a newly established university, which only has two science departments as yet).

For the purposes of this document, a query plan is simply a selection from the available set of sources for each role and concept name in the query; only the selected sources are to be used to answer the query. Other, lower-level, choices (such as the order in which sources are accessed) may be made in the query-planning process, but we assume that such choices do not affect the set of concept instances which are returned by the plan, and so do not affect soundness or completeness. More formally, a *plan* P for an $\mathcal{ALCQI}$ query Q is an $\mathcal{ALCQI}$ expression in which the atomic concepts and roles in Q have been replaced by unions of smaller subconcepts and subroles:

Definition 1. Let Q be a query (that is, an $\mathcal{ALCQI}$ expression). A *plan* P for Q is a query that can be obtained from Q by substituting a concept expression C' for each occurrence of an atomic concept C in Q, and a role R' for each occurrence of an atomic role R in Q, subject to the following restrictions:

- For each C, C' is a union $C'_1 \sqcup \cdots \sqcup C'_k$, where each C'_k is an atomic concept which is subsumed by C.
- For each R, R' is an atomic role which is subsumed by R.

For example, the query Q_1 defined above admits a plan $P_1 = \mathsf{hall\text{-}std} \sqcup \mathsf{cs\text{-}std}$, Q_2 admits a plan $P_2 = \mathsf{hall\text{-}std} \sqcap \neg\mathsf{cs\text{-}std}$ and Q_3 admits a plan $P_3 = \mathsf{hall\text{-}std} \sqcap \exists^{\leqslant 1}\mathsf{flat\text{-}mate.cs\text{-}std}$. These plans are all obtained by omitting the source phys-std for the concept science-std, perhaps because the Physics Department's database is currently off-line.

Our notions of soundness and completeness are analogous to the notions of precision and recall, respectively in the field of Information Retrieval. The *soundness* of a plan P with respect to a given query Q, denoted $\mathcal{S}(P, Q)$, is a measure of how many of the answers it produces are in fact members of the set of instances defined by the query. Although the definition given below depends on the contents of the databases, we will suppress the database state from the notation, and take it as read that we have in mind a particular database state.

Definition 2. Let P be a plan for a query Q. The *soundness* of P for Q, denoted $\mathcal{S}(P, Q)$ is:

$$\mathcal{S}(P, Q) = \#(P \sqcap Q)/\#(P). \tag{1}$$

Note that soundness always takes values between 0 and 1. Furthermore, the soundness of an empty plan (one which produces no answers) is not defined by the above formula. Since an empty plan clearly produces no wrong answers, we adopt the convention that the soundness of such a plan is 1.

Similarly, the *completeness* $\mathcal{C}(P,Q)$ of a plan P for a query Q is a measure of how many of the answers to Q are actually found by P.

Definition 3. Let P be a plan for a query Q. The *completeness* of P for Q, denoted $\mathcal{C}(P,Q)$ is:

$$\mathcal{C}(P,Q) = \#(P \sqcap Q)/\#(Q). \tag{2}$$

Again, we note that completeness takes values in the range [0,1]. In this case, the formula cannot be applied when the *query* is empty. Since we cannot miss any answers to an empty query, we also assign a value of 1 in this case.

So we can measure (or estimate) the soundness and completeness of a plan if we can measure (or estimate) the cardinality of an arbitrary concept expression.

4. Estimating cardinalities of DL concept extensions

4.1. Propositional expressions

We begin by considering the propositional fragment of $\mathcal{ALCQI}$, where the only connectives used are $\sqcap$, $\sqcup$ and $\neg$. This fragment is equivalent to classical propositional logic. In order to compute the cardinality of any safe propositional expression, it suffices to know the cardinalities of all conjunctions of atomic concepts. We can obtain them from the databases in advance of any query processing. Since the number of such conjunctions is exponential in the number of atomic concepts, it is not generally feasible to collect and store the cardinalities of all of them, except in cases where most of the cardinalities are zero. For example, it might be that the concept atoms can be divided into small "clusters" such that only concepts in the same cluster can have non-empty intersections. In such cases, we must choose a data structure and lookup algorithm so that the empty cases can be detected efficiently. In the general case, we must estimate the cardinalities of atomic conjunctions. A study of one possible estimation technique is given in Section 6.

Given such input, we can compute the cardinalities of all safe propositional concept expressions by (recursively) using the equations

$$\#(C \sqcup D) = \#(C) + \#(D) - \#(C \sqcap D), \tag{3}$$

$$\#(C \sqcap \neg D) = \#(C) - \#(C \sqcap D). \tag{4}$$

It may be noted that for large concept expressions the application of Eqs. (3) and (4) could involve a large amount of computation. For example, calculating the cardinality of a union $\bigcup_{i=1}^{n} C_i$ of atomic concepts C_i by rule (3) involves a summation of $2^n - 1$ terms (one for each non-empty subset of $\{1,\ldots,n\}$):

$$\#\left(\bigcup_{i=1}^{n} C_i\right) = \sum_{\{i_1,\ldots,i_k\}\subseteq\{1,\ldots,n\}} (-1)^k \#(C_{i_1} \cap \cdots \cap C_{i_k}). \tag{5}$$

This formula, which is easily proved by induction on n, is the well-known inclusion–exclusion principle of combinatorics (see [26, pp. 178–179], for example). However, in practice it is likely to

Table 1
Cardinalities for example sources

Concept	Cardinality
hall-std	1000
cs-std	1000
phys-std	1000
hall-std ⊓ cs-std	500
hall-std ⊓ phys-std	100
cs-std ⊓ phys-std	0
hall-std ⊓ cs-std ⊓ phys-std	0

be the case that most of the intersections are empty. In such cases much of the recursion may be avoided by not evaluating the (necessarily zero) last term on the right-hand side of Eq. (3) if either of the first two terms is zero. Similarly, we should not evaluate the second term on the right-hand side of Eq. (4) if the first term is zero.

Suppose that, in the example application described in Sections 2 and 3, the cardinalities of the sources and their intersections are given by Table 1. To find the soundness and completeness of the plan P_1 with respect to the query Q_1, we need to compute the cardinalities of P_1, Q_1 and $P_1 \sqcap Q_1$. But in this case, we have $P_1 \sqcap Q_1 \equiv P_1$, since P_1 is subsumed by Q_1. In fact, this is always the case when the query expression contains no *negative* operators—that is no negations, universal quantifiers or upper-bound number restrictions ($\exists^{\leqslant n} R.C$). So we immediately have $\mathcal{S}(P_1, Q_1) = 1$. Now

$$\begin{aligned}\#(Q_1) &= \#(\text{hall-std} \sqcup \text{cs-std} \sqcup \text{phys-std}) \\ &= \#(\text{hall-std}) + \#(\text{cs-std}) + \#(\text{phys-std}) - \#(\text{hall-std} \sqcap \text{cs-std}) \\ &\quad - \#(\text{hall-std} \sqcap \text{phys-std}) - \#(\text{cs-std} \sqcap \text{phys-std}) \\ &\quad + \#(\text{hall-std} \sqcap \text{cs-std} \sqcap \text{phys-std}) = 2400\end{aligned}$$

and

$$\#(P_1) = \#(\text{hall-std} \sqcup \text{cs-std}) = \#(\text{hall-std}) + \#(\text{cs-std}) - \#(\text{hall-std} \sqcap \text{cs-std}) = 1500,$$

so that $\mathcal{C}(P_1, Q_1) = \#(P_1 \sqcap Q_1)/\#(Q_1) = \#(P_1)/\#(Q_1) = 1500/2400 = 0.625$.

Now consider the query Q_2 and the plan P_2. Since in this case the incomplete data is used for a concept which is under a negation, we have $P_2 \sqcap Q_2 \equiv Q_2$ so that $\mathcal{C}(P_2, Q_2) = 1$. For soundness, we have

$$\begin{aligned}\#(Q_2) &= \#(\text{hall-std} \sqcap \neg(\text{cs-std} \sqcup \text{phys-std})) \\ &= \#(\text{hall-std}) - \#(\text{hall-std} \sqcap (\text{cs-std} \sqcap \text{phys-std})) \\ &= \#(\text{hall-std}) - \#((\text{hall-std} \sqcap \text{cs-std}) \sqcup (\text{hall-std} \sqcap \text{phys-std})) \\ &= \#(\text{hall-std}) - \#(\text{hall-std} \sqcap \text{cs-std}) - \#(\text{hall-std} \sqcap \text{phys-std}) \\ &\quad + \#(\text{hall-std} \sqcap \text{cs-std} \sqcap \text{phys-std}) = 1000 - 500 - 100 + 0 = 400\end{aligned}$$

and

$$\#(P_2) = \#(\mathsf{hall\text{-}std} \sqcap \neg \mathsf{cs\text{-}std}) = \#(\mathsf{hall\text{-}std}) - \#(\mathsf{hall\text{-}std} \sqcap \mathsf{cs\text{-}std}) = 1000 - 500 = 500,$$

which gives $\mathcal{S}(P_2, Q_2) = \#(P_2 \sqcap Q_2)/\#(P_2) = \#(Q_2)/\#(P_2) = 400/500 = 0.8$.

4.2. Quantified expressions

In order to estimate the cardinalities of quantified formulae, we adopt a probabilistic approach. As in the case of propositional expressions, the data required for our calculations can (in principle) be obtained by running a finite (though possibly large) number of queries in advance of general query processing. However, we cannot expect to obtain precise cardinality results, even with precise input data, since the filler concept C in an expression such as $\exists R.C$ may be an arbitrary complex concept expression. The data we require concerns the distributions of the numbers of fillers for elements in the domain of each role. First, we assume that we have concept expressions domain(R) and range(R) for the domain and range of any role R. These should be unions of atomic concepts (in an application, the atomic concepts will be the concepts domains and ranges of the corresponding database relations). The best results are likely to be obtained if the expression range(R) accurately captures the actual range of filler values of R, rather than just the potential values. That is, if there are not too many instances y of R with no x such that (x, y) is an instance of R. For each $i \geqslant 0$, let q_i be the probability that an element of domain(R) has exactly i R-fillers (so $\sum_{i=0}^{\infty} q_i = 1$). That is,

$$q_i = (\#(\exists^{\geqslant i} R.\top) - \#(\exists^{\geqslant i-1} R.\top))/\#(\mathrm{domain}(R)). \tag{6}$$

As with the cardinalities of intersections of source concepts, we could evaluate the q_i for each role in advance of query processing, and update the values periodically. If the q_i are calculated directly from Eq. (6), then only finitely many of them are non-zero for each R. In fact, if R is single-valued then $q_i = 0$ for $i > 1$. In general, we require that this finiteness condition holds for any set of values which we use for the q_i. (In some situations, we might approximate the q_i by, say, a Poisson distribution with a given mean, suitably truncated.)

To estimate the cardinality of an existentially quantified formula, $\exists^{\geqslant n} R.C$, where $n \geqslant 1$, let $P(n, R, C)$ be the probability that an element of domain(R) has at least n R-fillers in C. Then

$$\#(\exists^{\geqslant n} R.C) = \#(\mathrm{domain}(R))P(n, R, C), \tag{7}$$

so we need to estimate $P(n, R, C)$.

Let p be the probability that any given R-filler is in C. We can estimate p by the formula

$$p \approx \#(\mathrm{range}(R) \sqcap C)/\#(\mathrm{range}(R)) \tag{8}$$

(we can assume that $\#(\mathrm{range}(R)) \neq 0$ since otherwise we have $P(n, R, C) = 0$ for any C). Note that the expression range(R) $\sqcap C$ is safe (even if C is unsafe) and contains fewer quantifiers than $\exists^{\geqslant n} R.C$ so, by induction, we can estimate its cardinality. Then

$$P(n, R, C) = \sum_{i=1}^{\infty} q_i \quad \text{(probability that at least } n \text{ out of } i \text{ fillers are in } C\text{)}. \tag{9}$$

Since only finitely many of the q_i are non-zero for each R, the sum in Eq. (9) is actually finite.

Now let us assume that the probabilities of distinct R-fillers (for a given object) being in C are independent of each other. Then, given that we have exactly i fillers, the probability that j of them

are in C is given by a binomial distribution. The probability that j out of i fillers are in C is $\binom{i}{j}p^j(1-p)^{i-j}$, where $\binom{i}{j}$ denotes the binomial coefficient $i!/(j!(i-j)!)$. This formula derives from the fact that there are $\binom{i}{j}$ j-element subsets of the i fillers and, for each subset, the probability that all of its elements and none of the other fillers are in C is $p^j(1-p)^{i-j}$. Then we can estimate $P(n,R,C)$ by $Q(n,R,C)$ where

$$Q(n,R,C) = \sum_{i=1}^{\infty} q_i \sum_{j=n}^{i} \binom{i}{j} p^j (1-p)^{i-j}. \tag{10}$$

Note that when $i \geqslant 2n$, the calculation of the inner sum can be simplified by using the identity

$$\sum_{j=n}^{i} \binom{i}{j} p^j (1-p)^{i-j} = 1 - \sum_{j=0}^{n-1} \binom{i}{j} p^j (1-p)^{i-j}, \tag{11}$$

where the sum on the right-hand side has fewer terms than the one on the left. (Eq. (11) is easily derived from the fact that the two summations taken together form the binomial expansion of $(p+(1-p))^i$). In particular, if $n=1$ (so we are considering the concept $\exists R.C$), we can compute $Q(1,R,C)$ as

$$Q(1,R,C) = \sum_{i=1}^{\infty} q_i (1-(1-p)^i). \tag{12}$$

In general, if q_i is non-zero for large i we can approximate the inner sum in Eq. (10) for large values of i by using a normal (Gaussian) distribution with mean $\mu = ip$ and standard deviation $\sigma = (ip(1-p))^{1/2}$:

$$\sum_{j=n}^{i} \binom{i}{j} p^j (1-p)^{i-j} \approx \frac{1}{\sqrt{2\pi}} \int_{(n-\mu)/\sigma}^{\infty} \mathrm{e}^{-(1/2)x^2}\,\mathrm{d}x. \tag{13}$$

The value of this integral can be approximated by interpolation in standard statistical tables.

So we can estimate $\#(\exists^{\geqslant n} R.C)$ by the formula

$$\#(\exists^{\geqslant n} R.C) \approx \#(\mathrm{domain}(R))Q(n,R,C). \tag{14}$$

More generally, let D be a concept expression, and suppose we wish to estimate $\#(D \sqcap \exists^{\geqslant n} R.C)$. If we assume that both the distribution of numbers of R-fillers (the q_i defined above) and the probability p that a given R-filler is in C are statistically independent of membership of D, we have the approximation

$$\#(D \sqcap \exists^{\geqslant n} R.C) \approx \#(D \sqcap \mathrm{domain}(R))Q(n,R,C). \tag{15}$$

Note that the expression $D \sqcap \mathrm{domain}(R)$ is safe, and therefore we can estimate its cardinality, since the form of the expression for $\mathrm{domain}(R)$ specified in Section 4.2 is safe.

Similarly, suppose we wish to estimate $\#(D \sqcap \exists^{\geqslant n_1} R_1.C_1 \sqcap \exists^{\geqslant n_2} R_2.C_2)$. We assume as above that the values of p or the q_i for both R_1 and R_2 are independent of membership in D, and that the values for R_1 and R_2 are independent of each other. Then we use the approximation

$$\#(D \sqcap \exists^{\geqslant n} R_1.C_1 \sqcap \exists^{\geqslant n} R_2.C_2) \approx \#(D \sqcap \mathrm{domain}(R_1) \sqcap \mathrm{domain}(R_2)) Q(n_1, R_1, C_1) \times Q(n_2, R_2, C_2) \tag{16}$$

and so on.

By combining rules like (15) and (16) with rules (3) and (4) we can estimate the cardinality of any safe $\mathcal{ALCQI}$ concept expression. A complete algorithm is given in Appendix A.

Suppose that, in our example application, the data we need concerning the role flat-mate are given by Table 2. Consider the plan P_3 for query Q_3. As with the example involving Q_2, we have $P_3 \sqcap Q_3 \equiv Q_3$ and so $\mathcal{C}(P_3, Q_3) = 1$. Since, in general, $\exists^{\leqslant n} R.C \equiv \neg\exists^{\geqslant n+1} R.C$, we have

$$\begin{aligned}\#(Q_3) &= \#(\mathsf{hall\text{-}std} \sqcap \neg\exists^{\geqslant 2}\mathsf{flat\text{-}mate}.(\mathsf{cs\text{-}std} \sqcup \mathsf{phys\text{-}std})) \\ &= \#(\mathsf{hall\text{-}std}) - \#(\mathsf{hall\text{-}std} \sqcap \exists^{\geqslant 2}\mathsf{flat\text{-}mate}.(\mathsf{cs\text{-}std} \sqcup \mathsf{phys\text{-}std})) \\ &\approx \#(\mathsf{hall\text{-}std}) - \#(\mathsf{hall\text{-}std} \sqcap \mathrm{domain}(\mathsf{flat\text{-}mate}))Q(2, \mathsf{flat\text{-}mate}, \mathsf{cs\text{-}std} \sqcup \mathsf{phys\text{-}std}).\end{aligned}$$

To evaluate $Q = Q(2, \mathsf{flat\text{-}mate},\ \mathsf{cs\text{-}std} \sqcup \mathsf{phys\text{-}std})$, we need to compute

$$\begin{aligned}p &= \frac{\#(\mathrm{range}(\mathsf{flat\text{-}mate}) \sqcap (\mathsf{cs\text{-}std} \sqcup \mathsf{phys\text{-}std}))}{\#(\mathrm{range}(\mathsf{flat\text{-}mate}))} \\ &= \frac{(\#(\mathrm{range}(\mathsf{flat\text{-}mate}) \sqcap \mathsf{cs\text{-}std}) + \#(\mathrm{range}(\mathsf{flat\text{-}mate}) \sqcap \mathsf{phys\text{-}std}) - \#(\mathrm{range}(\mathsf{flat\text{-}mate}) \sqcap \mathsf{cs\text{-}std} \sqcap \mathsf{phys\text{-}std}))}{\#(\mathrm{range}(\mathsf{flat\text{-}mate}))} \\ &= (600 + 400 - 0)/1500 \approx 0.67.\end{aligned}$$

Then

$$\begin{aligned}Q &= q_2\binom{2}{2}p^2 + q_3\left(\binom{3}{2}p^2(1-p) + \binom{3}{3}p^3\right) + q_4\left(1 - \binom{4}{0}(1-p)^4 - \binom{4}{1}p(1-p)^3\right) \\ &= 0.18(1(4/9)) + 0.08(3(4/27) + 1(8/27)) + 0.02(1 - 1(1/81) - 4(2/81)) \approx 0.16,\end{aligned}$$

Table 2
Cardinality data for the example role flat-mate

Domain and range	
Concept	Cardinality
hall-std ⊓ domain(flat-mate)	800
range(flat-mate)	1500
cs-std ⊓ range(flat-mate)	600
phys-std ⊓ range(flat-mate)	400
cs-std ⊓ phys-std ⊓ range(flat-mate)	0
i	q_i
Filler count distribution	
0	0.35
1	0.37
2	0.18
3	0.08
4	0.02
⩾5	0

so our approximation to $\#(Q_3)$ is

$$\#(Q_3) \approx 1000 - 800 \times 0.16 \approx 870.$$

For $\#(P_3)$ we make a similar calculation for $Q(2, \mathsf{flat\text{-}mate}, \mathsf{cs\text{-}std})$, except that we use the value

$$\frac{\#(\mathrm{range}(\mathsf{flat\text{-}mate}) \sqcap \mathsf{cs\text{-}std})}{\#(\mathrm{range}(\mathsf{flat\text{-}mate}))} = 600/1500 = 0.4$$

for p. This yields the value 0.056. So

$$\begin{aligned}\#(P_3) &= \#(\mathsf{hall\text{-}std} \sqcap \neg \exists^{\geqslant 2} \mathsf{flat\text{-}mate.cs\text{-}std}) \\ &= \#(\mathsf{hall\text{-}std}) - \#(\mathsf{hall\text{-}std} \sqcap \exists^{\geqslant 2} \mathsf{flat\text{-}mate.cs\text{-}std}) \\ &\approx \#(\mathsf{hall\text{-}std}) - \#(\mathsf{hall\text{-}std} \sqcap \mathrm{domain}(\mathsf{flat\text{-}mate}))Q(2, \mathsf{flat\text{-}mate}, \mathsf{cs\text{-}std}) \\ &\approx 1000 - 800 \times 0.056 \approx 960.\end{aligned}$$

So our estimate of the soundness is

$$\mathcal{S}(P_3, Q_3) = \frac{\#(P_3 \sqcap Q_3)}{\#(P_3)} = \frac{\#(Q_3)}{\#(P_3)} \approx \frac{870}{960} \approx 0.91.$$

5. Statistical evaluation of estimation methods

In order to validate the method proposed in Section 4.2 for estimating cardinalities of quantified expressions and, by extension, for estimating soundness and completeness of evaluation plans for such expressions, we have run some statistical tests using randomly generated data.

We consider the expression $\exists R.C$, which is equivalent to $\exists^{\geqslant 1} R.C$. For given values of $d = \#(\mathrm{domain}(R))$, $r = \#(\mathrm{range}(R))$, and $s = \#(R)$ the test system generates a random s-element subset of $\{1, \ldots, d\} \times \{1, \ldots, r\}$, to represent the instances of R. This is done by using uniformly distributed random numbers for the domain and range values until s distinct pairs have been generated. The system also generates two "sources" C_1 and C_2 for the filler relation C by generating two random subsets of $\{1, \ldots, r\}$ with specified cardinalities c_1 and c_2. We then use our estimation techniques to compute estimates of $\#(C_1)$ and of the completeness $\mathcal{C}(C_1, C)$ of C_1 with respect to $C = C_1 \sqcup C_2$. We have examined the behaviour of our estimation techniques for varying values of the specified cardinalities.

Table 3 indicates the accuracy of our cardinality estimations. It contains data for varying values of $\#(C_1)$ for the case where $\#(\mathrm{domain}(R))$, $\#(\mathrm{range}(R))$ and $\#(R)$ have all been fixed at 100. Results are shown for 1000 trials for each value of $\#(C_1)$. The table shows the mean of the actual values of $\#(\exists R.C_1)$. As a measure of the accuracy of our methods in estimating $\#(\exists R.C_1)$, the table shows a *normalised error* value. The normalised error is calculated by taking the root mean square (RMS) of the differences between the estimated and true values and dividing by the mean of all the true values, so that it represents a relative, rather than absolute, error.

Table 4 shows results for completeness estimations, using the same generated data as in Table 3 with the addition of a set of instances for C_2, which in these examples always has cardinality 50. The table shows the mean of the true values of the completeness $\mathcal{C}(\exists R.C_1, \exists R.C)$ and the RMS deviation of the estimated value from the true value.

We observe from these tables (and from further test results which have been omitted to save space) that our estimation methods appear to be reasonably successful in cases where the

Table 3
Cardinality estimation for 1000 trials with $\#(\text{domain}(R)) = \#(\text{range}(R)) = \#(R) = 100$

$\#(C_1)$	$\#(C_2)$	Mean of $\#(\exists R.C_1)$	Normalised error
10	50	9.42	0.29
20	50	18.26	0.18
30	50	25.78	0.14
40	50	33.18	0.11
50	50	39.47	0.09
60	50	45.20	0.07
70	50	50.62	0.06
80	50	55.36	0.05
90	50	59.59	0.03

Table 4
Completeness estimation with $\#(\text{domain}(R)) = \#(\text{range}(R)) = \#(R) = 100$

$\#(C_1)$	$\#(C_2)$	Mean completeness	RMS error
10	50	0.26	0.07
20	50	0.40	0.07
30	50	0.53	0.07
40	50	0.65	0.07
50	50	0.75	0.06
60	50	0.82	0.05
70	50	0.88	0.04
80	50	0.92	0.03
90	50	0.97	0.02

cardinalities are not too small, but tend to break down for small answer sets. This is what one might expect. For example, if the cardinality of a set is estimated as 0.2, then in most cases the true figure will be either 0 or 1, and which one it is will make a difference which is more significant than the difference between, say 99 and 100. In the context of our query-processing system, we may be able to improve the accuracy of our soundness and completeness estimates by evaluating subqueries which we estimate to have a small number of elements. If the access times for the sources involved in the subqueries have a latency which is not too high, we will be able to run such subqueries quickly in most cases. The size threshold below which we would use this technique would be determined by experiment.

6. Estimating conjunction cardinalities

The cardinality estimation method described in Section 4 requires, as input, values for the cardinalities of all possible conjunctions of atomic concepts. In an application with many potentially overlapping concepts or where answering queries to obtain these conjunction cardinalities is expensive, it may be impractical to gather and store all this data. In such cases it is necessary to find a means of estimating these basic cardinalities.

As a simple example of such an estimation technique, suppose we estimate the cardinality of an intersection $A_1 \sqcap \cdots \sqcap A_n$ of distinct atomic concepts by

$$\#(A_1 \sqcap \cdots \sqcap A_n) \approx \frac{\min_i(\#(A_i))}{n}. \tag{17}$$

This method is unlikely to give us very accurate results for the cardinalities themselves, but in some applications it might serve to give a reasonable estimate of whether a query plan has acceptable levels of soundness and completeness.

Tables 5–9 show some results of our completeness estimation algorithm, where the intersection cardinalities are estimated by using Eq. (17). The tests were performed by drawing evenly distributed random subsets from the set $\{1, \ldots, 1000\}$ with specified cardinalities as shown in the tables. The tables show completeness estimates and mean completeness values for $\mathcal{C}(P, Q)$ when using the plan $P = A \sqcap B$ to answer the query $Q = A \sqcap (B_1 \sqcup B_2)$. In each case 1000 trials were performed. Since the concept sizes used in the tests were fairly large, the standard deviations of the completeness values were quite small in all cases, and they are not shown in the tables.

Table 5 shows results for $\#(A) = 500$ and $\#(B_1) = \#(B_2) = k$. In each case, $\mathcal{C}(P, Q)$ is estimated as 3/4, since $\#(P \sqcap Q)$ is estimated as $\min(k, 500)/2$ and $\#(Q)$ as $2\min(k, 500)/3$. The actual completeness is lower than estimated when k is small compared to the range size of 1000, and

Table 5
Results for $\#(A) = 500$, $\#(B_1) = \#(B_2)$

$\#(A)$	$\#(B_1)$	$\#(B_2)$	Estimated completeness	Mean completeness
500	100	100	0.75	0.53
500	200	200	0.75	0.56
500	300	300	0.75	0.59
500	400	400	0.75	0.63
500	500	500	0.75	0.67
500	600	600	0.75	0.71
500	700	700	0.75	0.77
500	800	800	0.75	0.83
500	900	900	0.75	0.91

Table 6
Results for $\#(A) = \#(B_1) = \#(B_2)$

$\#(A)$	$\#(B_1)$	$\#(B_2)$	Estimated completeness	Mean completeness
100	100	100	0.75	0.52
200	200	200	0.75	0.55
300	300	300	0.75	0.59
400	400	400	0.75	0.63
500	500	500	0.75	0.67
600	600	600	0.75	0.71
700	700	700	0.75	0.77
800	800	800	0.75	0.83
900	900	900	0.75	0.91

Table 7
Results for $\#(A) = \#(B_1) = 500$

$\#(A)$	$\#(B_1)$	$\#(B_2)$	Estimated completeness	Mean completeness
500	500	100	0.94	0.91
500	500	200	0.88	0.83
500	500	300	0.83	0.77
500	500	400	0.79	0.71
500	500	500	0.75	0.67
500	500	600	0.75	0.63
500	500	700	0.75	0.59
500	500	800	0.75	0.56
500	500	900	0.75	0.53

Table 8
Results for $\#(A) = 500$, $\#(B_1) = 100$

$\#(A)$	$\#(B_1)$	$\#(B_2)$	Estimated completeness	Mean completeness
500	100	100	0.75	0.53
500	100	200	0.43	0.36
500	100	300	0.30	0.27
500	100	400	0.23	0.22
500	100	500	0.19	0.18
500	100	600	0.19	0.16
500	100	700	0.19	0.14
500	100	800	0.19	0.12
500	100	900	0.19	0.11

Table 9
Results for $\#(A) = 500$, $\#(B_1) = 900$

$\#(A)$	$\#(B_1)$	$\#(B_2)$	Estimated completeness	Mean completeness
500	900	100	0.94	0.99
500	900	200	0.88	0.98
500	900	300	0.83	0.97
500	900	400	0.79	0.96
500	900	500	0.75	0.95
500	900	600	0.75	0.94
500	900	700	0.75	0.93
500	900	800	0.75	0.92
500	900	900	0.75	0.91

higher when k is large. This is due to the fact that there will be more overlap between B_1 and B_2 (and hence less loss of completeness) when they are more densely distributed within their range of possible values. Table 6 shows results for $\#(A) = \#(B_1) = \#(B_2)$. The completeness figures are very similar to those in Table 5, which suggests that the results for $\mathcal{C}(P, Q)$ are not particularly sensitive to variations in $\#(A)$.

Table 7 shows results for $\#(A) = \#(B_1) = 500$, for varying values of $\#(B_2)$. As one would expect, the actual completeness values decrease as the size of the omitted extent B_2 increases. The estimated values show the same general trend, but stop decreasing once $\#(B_2)$ becomes greater than $\#(A) = 500$. This is because we estimate $\#(P \sqcap Q)$ as 250 and $\#(Q)$ as $250 + \min(500, \#(B_2))/6$. Tables 8 and 9 also show results for varying $\#(B_2)$, but with $\#(B_1)$ fixed at 100 and 900 respectively. Here again the estimates follow the general decreasing trend as the true values, but stop decreasing when $\#(B_2)$ reaches the same value as $\#(A)$.

7. Interactive improvement of soundness and completeness

Having formulated a query, the user of the system must choose a set of concept and role sources to be used in a plan for answering that query. The user will then be given soundness and completeness estimates for the plan. If he/she is not satisfied with either the soundness or completeness of the plan, the system should be able to provide some assistance in selecting additional sources which will be of most benefit in improving the plan's expected quality. If adding new sources causes problems with cost estimates, the user may also require some assistance in selecting sources for removal from the plan without causing too much harm to the quality estimates. We present below a set of heuristic measures for analysing a query in order to provide such assistance, and some suggestions for how these measures might be presented in the user interface.

If queries are rewritten in terms of source concepts and roles by a simple global-as-view expansion process, as in [9], we can present the query to the user in its original, unexpanded form and expand concept names in subqueries as required when the user chooses to investigate particular subqueries. For example, if the current query contains the concept biopolymer which has the expansion protein $\sqcup$ dna, the user would have the option of treating biopolymer as an atomic concept, whose sources are the union of all protein and dna sources, or of expanding and treating the disjuncts separately. If a more sophisticated rewriting system is used, as in DWQ [2] or PICSEL [4], it might be necessary for the user to work with the rewritten form of the query, as there may be little or no relation between the syntactic forms of the user's query and the plan.

7.1. Improving disjunctive expressions

If a disjunctive query is safe, then each disjunct must be safe. So we can assume our query is $Q = Q_1 \sqcup \cdots \sqcup Q_n$ where each Q_i is safe. Then our plan can be expressed as $P = P_1 \sqcup \cdots \sqcup P_n$ where each P_i is a plan for Q_i.

In order to improve the completeness of Q, we need to identify disjuncts for which adding new sources is likely to generate the most new answers. So we should look for disjuncts such that $\#(Q_i)$ is large and $\mathcal{C}(P_i, Q_i)$ is small. As a numerical measure of how promising each disjunct Q_i is for improving completeness, we can use the quantity $\#(Q_i)(1 - \mathcal{C}(P_i, Q_i))$ or, equivalently, $\#(Q_i) - \#(P_i \sqcap Q_i)$. We can display these quantities to the user in the form of a histogram showing, for each disjunct either the estimated values of $\#(Q_i)$ and $\#(P_i \sqcap Q_i)$ for each i, as in Fig. 2, or just the differences, as in Fig. 3. The query (or subquery) under investigation in the figures is science-std $\sqcup$ engineering-std $\sqcup$ business-std. In both cases, we have indicated the scale of the graph by displaying the greatest of the cardinalities involved.

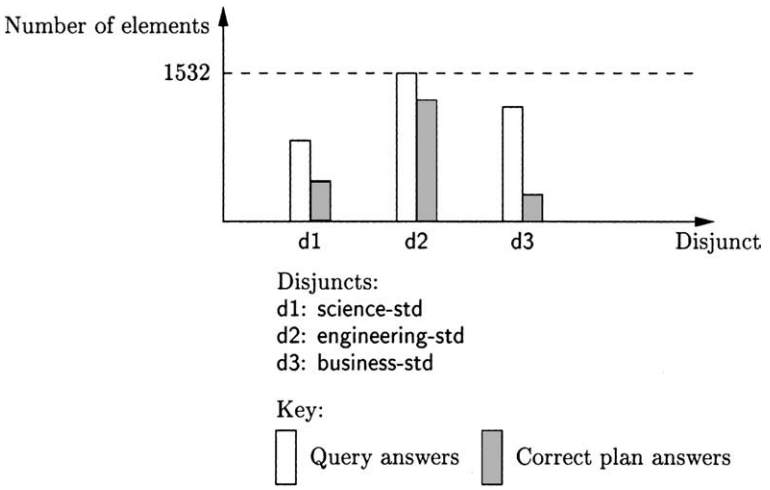

Fig. 2. Histogram for comparison of completeness of disjuncts, showing (estimated) sizes of complete answers and current plan.

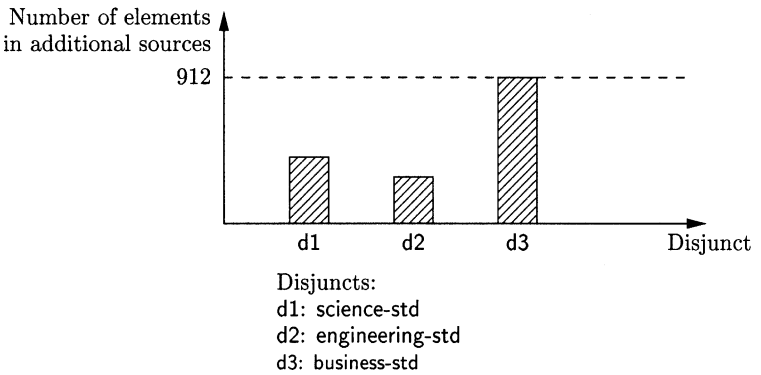

Fig. 3. Alternative completeness histogram showing numbers of extra elements available for each disjunct.

Similarly, to improve the soundness of a plan we look for disjuncts where adding new sources is likely to remove the greatest number of wrong answers (by improving the completeness of subqueries which lie under an odd number of negations). So we look for disjuncts such that $\#(P_i)$ is large and $\mathcal{S}(P_i, Q_i)$ is small. As a numerical measure, we can take $\#(P_i)(1 - \mathcal{S}(P_i, Q_i))$ or, equivalently $\#(P_i) - \#(P_i \sqcap Q_i)$. The information can be displayed to the user in graphs similar to those in Figs. 2 and 3. Fig. 4 illustrates the former option for the query science-std $\sqcup$ engineering-std $\sqcup$ business-std.

Of course, the numerical measures suggested above ignore possible overlaps between disjuncts. For example, suppose $Q = Q_1 \sqcup Q_2 \sqcup Q_3$. If $\mathcal{C}(P_1, Q_1)$ is close to 1 and Q_1 and Q_2 have a high proportion of elements in common, then it may more productive to improve $\mathcal{C}(P_3, Q_3)$ than $\mathcal{C}(P_2, Q_2)$, since any new elements generated by P_2 may well be covered by P_1. However, a more

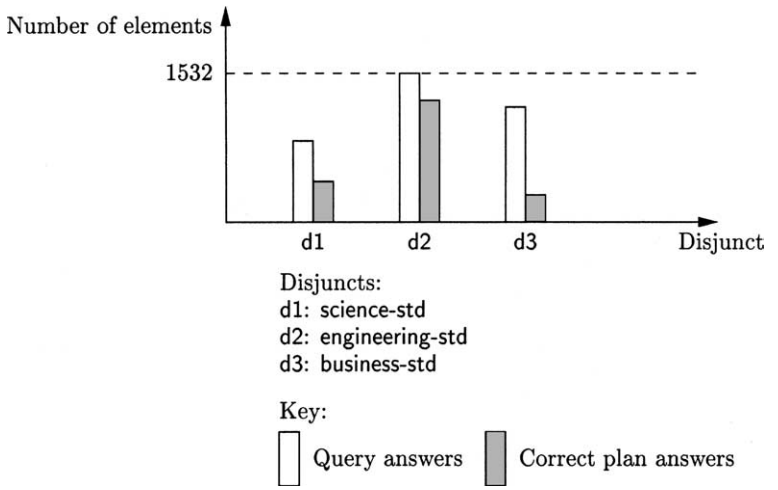

Fig. 4. Histogram for comparison of soundness of disjuncts, showing (estimated) sizes of current plan answers and correct plan answers.

sophisticated approach which took overlaps into account would have to be based on better estimates or data for intersection cardinalities than those given in Section 6.

7.2. Improving conjunctive expressions

A safe conjunction has the form $Q = S_1 \sqcap \cdots \sqcap S_m \sqcap \cdots \sqcap U_1 \sqcap \cdots \sqcap U_n$, where $m \geqslant 1$, $n \geqslant 0$, each S_i is safe and each U_i is unsafe. We answer such a query (at least in principle) by answering each safe conjunct S_i and the negation $\neg U_i$ of each unsafe conjunct. We then filter the elements of the intersection of the S_i for non-membership in any of the $\neg U_i$.

To improve the completeness of a plan P for Q, we can either improve the completeness of some of the S_i or the soundness of the negation of one of the U_i (so that we do not incorrectly filter any elements out). Unlike the case of disjunctions (Section 7.1) a lack of completeness in any of the safe conjuncts can have a large impact on the completeness of the overall query, even if that conjunct has low cardinality. It therefore seems appropriate to use the completeness/soundness values themselves as a numerical indication of which conjuncts should be considered for improvement. The information about the various conjuncts could be presented to the user in a graph similar to Fig. 5, where the query (or subquery) under examination is hall-std $\sqcap$ science-std $\sqcap$ $\neg$phys-std (so $m = 2$ and $n = 1$). Alternatively, it might be better to hide the distinction between safe and unsafe conjuncts from the user by presenting them in the same way in this part of the interface so that we refer to the "completeness" of, say, $\neg$phys-std or $\forall$flat-mate.science-std rather than soundness of phys-std or $\exists$flat-mate.$\neg$science-std even though the unsafe versions will never actually be evaluated.

As it was with disjunctive queries, the problem of improving soundness of a query is dual to the problem of improving completeness. We can improve the soundness of a conjunction either by improving the soundness of a safe conjunct or by improving the completeness of the negation of an unsafe conjunct.

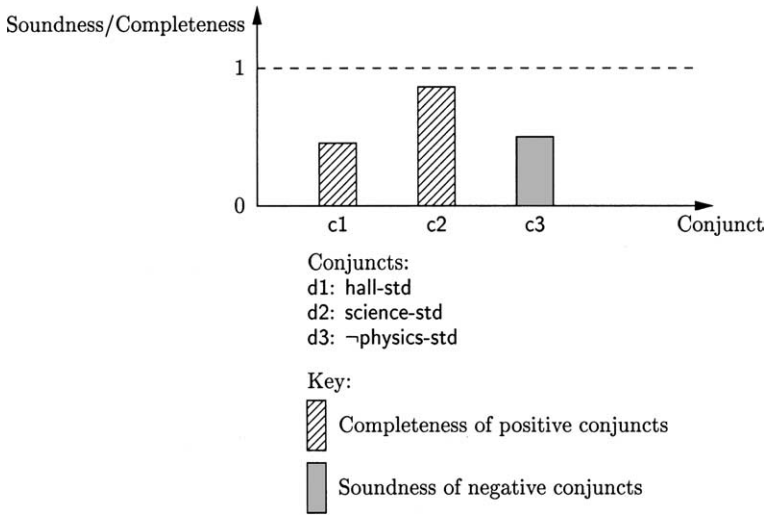

Fig. 5. Histogram for comparison of completeness of conjuncts.

7.3. Improving quantified expressions

A safe quantified expression is equivalent to one of the form $\exists \geqslant^n R.C$. The concept expression C may be either safe or unsafe. To improve the completeness of $\exists \geqslant^n R.C$, we can either improve the "completeness" of R (that is, add more sources for R) or improve the completeness of C (if C is safe) or the soundness of $\neg C$ (if C is unsafe). To improve soundness, we can improve the soundness of C (if C is unsafe) or the completeness of $\neg C$ (if C is safe). Note that, since $\mathcal{ALCQI}$ allows only atomic role expressions and their inverses, the use of restricted information for R has no impact on the soundness of a plan for $\exists \geqslant^n R.C$. In the case of completeness, it is not easy to see how the statistical information about the role R which is used in cardinality estimation (Section 4) could be used to decide whether improving the plan for R or the plan for C is likely to be more effective. It may be best simply to point out both possibilities to the user.

8. Conclusions

DL based global conceptual models have been widely proposed for use in information integration systems. However, it is not always practical or possible for queries over such models to access all the extensional data in remote sources. Thus it is important that query-processing environments are able to provide indications to users of the consequences for the soundness and completeness of results when specific sources are omitted. The methods presented in this paper represent a direct approach to providing such indications, and we have offered some suggestions as to how they should be presented to users in order to assist them in source selection. The soundness and completeness measures considered here are just two examples of data quality di-

mensions. Our work represents a first step in considering data quality issues in a DL based integration setting. Compared with earlier work on data quality estimation (for example [19]), the results presented here:

- act over a DL-based query language;
- include evaluation of the results obtained;
- address how quality measures can be conveyed to users.

Query cardinality estimation is a well-studied area in database research, as it is an important part of cost estimation in query optimisation (see [10], for example). However, we are not aware of any previous work on cardinality or quality estimation for expressive DL queries.

The results of the statistical validation tests (Section 5) suggest that our methods give good results in cases where the cardinality of the result is not too small, at least in the case of random data. Further tests, against real data and ontologies, will be made as soon as the system described in [9] has been fully implemented. Further work which could be done to improve our estimation technique might involve the use of domain-specific or application-specific knowledge to replace the many statistical independence assumptions made in the estimation process. One such approach might be to investigate the incorporation of statistical/probabilistic information about relations between cardinalities of concepts into the ontology, along the lines described in [27], to see whether this information could be used to improve our cardinality estimates. Another technique for improving estimates would be to replace the simple approximation method for atomic conjunctions in Section 6 by a more sophisticated one, which might, for example, use statistics gathered while answering previous queries.

Acknowledgements

This work has been supported by the UK Engineering and Physical Science Research Council (EPSRC), whose support we are pleased to acknowledge.

Appendix A. Cardinality estimation algorithm

In this appendix, we present an algorithm which can be used to estimate concept cardinalities in accordance with the approach described in Section 4. The algorithm begins with rewriting the concept expression into an equivalent expression in disjunctive normal form (DNF)—that is, $e = (c_{1,1} \sqcap \cdots \sqcap c_{1,m_1}) \sqcup \cdots \sqcup (c_{n,1} \sqcap \cdots \sqcap c_{n,m_n})$, where each $c_{i,j}$ is an atom A (safe), a negated atom $\neg A$ (unsafe), a quantified expression $\exists^{\geqslant k} R.C$, $k \geqslant 1$ (safe) or a quantified expression $\exists^{\leqslant k} R.C$, $k \geqslant 0$ (unsafe). In the case of quantified expressions, C should itself be in DNF. A DNF expression is safe if and only if at least one conjunct in each disjunct is safe. It is a straightforward matter to show that any $\mathcal{ALCQI}$ expression has an equivalent expression in DNF, and that the expression is safe if and only if its DNF equivalent is safe. The rewriting step may be computationally expensive, however, and other algorithms may be possible which avoid this step.

The function *dnf-estimate* returns a cardinality estimate for a safe $\mathcal{ALCQI}$ expression e in DNF.

function *dnf-estimate*(safe-dnf-expression e)
  if $e = \bot$
    return 0
  else if e has only one disjunct, d
    return *conj-estimate*(d)
  else
    let $d = d_1 \sqcup d_2 \sqcup \cdots \sqcup d_n$, $n \geqslant 2$
     $n_1 \leftarrow$ *conj-estimate*(d_1)
     $n_2 \leftarrow$ *dnf-estimate*$(d_2 \sqcup \cdots \sqcup d_n)$
     if $n_1 = 0$
       return n_2
     else if $n_2 = 0$
       return n_1
     else
       return $n_1 + n_2 -$ *dnf-estimate*$((d_1 \sqcap d_2) \sqcup \cdots \sqcup (d_1 \sqcap d_n))$

The function *conj-estimate* returns a cardinality estimate for a single disjunct of a safe DNF expression—a conjunction of terms as described above, at least one of which is safe.

function *conj-estimate*(safe-conjunction e)
  let $e = S_1 \sqcap \cdots \sqcap S_k \sqcap U_1 \sqcap \cdots \sqcap U_l$, S_i safe, U_j unsafe, $k \geqslant 1$
    if $l = 0$
     return *safe-conj-estimate*(e)
    else
     $n \leftarrow$ *conj-estimate*$(S_1 \sqcap \cdots \sqcap S_k \sqcap U_2 \sqcap \cdots \sqcap U_l)$
     if $n = 0$
       return 0
     else
       return $n -$ *conj-estimate*$(S_1 \sqcap \cdots \sqcap S_k \sqcap (\neg U_1) \sqcap U_2 \sqcap \cdots \sqcap U_l)$

The function *safe-conj-estimate* returns a cardinality estimate for a non-empty conjunction of safe terms. Here $Q(k_i, R_i, C_i)$ is the function defined in Section 4.2. Note that evaluating Q involves recursive calls to *dnf-estimate* to evaluate the right-hand side of Eq. (8).

function *conj-estimate*(all-safe-conjunction e)
  let $e = A_1 \sqcap \cdots \sqcap A_m \sqcap \exists^{\geqslant k_1} R_1.C_1 \sqcap \cdots \sqcap \exists^{\geqslant k_n} R_n.C_n$
    $x \leftarrow$ *lookup-card*$(A_1 \sqcap \cdots \sqcap A_m \sqcap \mathrm{domain}(R_1) \sqcap \cdots \sqcap \mathrm{domain}(R_n))$
    if $x = 0$
     return 0
    else
     return $x * Q(k_1, R_1, C_1) * \cdots * Q(k_n, R_n, C_n)$

References

[1] C.A. Goble, R. Stevens, G. Ng, S. Bechhofer, N.W. Paton, P.G. Baker, M. Peim, A. Brass, Transparent access to multiple bioinformatics information sources, IBM Systems Journal 40 (2) (2001) 532–551.

[2] D. Calvanese, G.D. Giacomo, M. Lenzerini, D. Nardi, R. Rosati, Information integration: conceptual modeling and reasoning support, in: Proceedings of the 3rd IFCIS International Conference on Cooperative Information Systems, IEEE-CS Press, 1998, pp. 280–291.

[3] A.Y. Levy, D. Srivastava, T. Kirk, Data model and query evaluation in global information systems, Journal of Intelligent Information Systems 5 (2) (1995) 121–143.

[4] F. Goasdoué, V. Lattes, M.-C. Rousset, The use of CARIN language and algorithms for information integration: the PICSEL system, International Journal of Cooperative Information Systems 9 (4) (2000) 383–401.

[5] Y. Arens, C.A. Knoblock, W.-M. Shen, Query reformulation for dynamic information integration, Journal of Intelligent Information Systems 6 (2/3) (1996) 99–130.

[6] S. Bechhofer, R. Stevens, G. Ng, A. Jacoby, C.A. Goble, Guiding the user: an ontology driven interface, in: N.W. Paton, T. Griffiths (Eds.), 1999 User Interfaces to Data Intensive Systems (UIDIS'99), IEEE Computer Society, 1999, pp. 158–161.

[7] P. Bresciani, P. Fontana, A knowledge-based query system for biological databases, in: T. Andreasen, A. Motro, H. Christiansen, H.L. Larsen (Eds.), Flexible Query Answering Systems, 5th International Conference, FQAS 2002, Copenhagen, Denmark, October 27–29, 2002, Proceedings, Lecture Notes in Computer Science, vol. 2522, Springer, 2002, pp. 86–99.

[8] J.D. Ullman, Information integration using logical views, in: Proceedings of the 6th International Conference on Database theory (ICDT'97), Lecture Notes in Computer Science, vol. 1186, Springer-Verlag, 1997, pp. 19–40.

[9] M. Peim, E. Franconi, N.W. Paton, C.A. Goble, Query processing with description logic ontologies over object-wrapped databases, in: J. Kennedy (Ed.), 14th International Conference on Scientific and Statistical Database Management, IEEE Computer Society, 2002, pp. 27–36.

[10] H. Garcia-Molina, J.D. Ullman, J. Widom, Database System Implementation, Prentice Hall, 2000.

[11] F. Najjar, Y. Slimani, Cardinality estimation of distributed join queries, in: Proceedings of the 10th DEXA Workshop, IEEE, 1999, pp. 66–70.

[12] R.Y. Wang, D.M. Strong, Beyond accuracy: What data quality means to data consumers, Journal of Management Information Systems 12 (4) (1996).

[13] T. Johnson, T. Dasu, Bellman: a data quality browser, in: Proceedings of the Sixth International Conference on Information Quality (IQ2001), Boston, MA, USA, 2001.

[14] D. Theodoratos, M. Bouzeghoub, Data currency quality satisfaction in the design of a data warehouse, IJCIS 10 (3) (2001) 299–326.

[15] R.Y. Wang, A product perspective on total data quality management, Communications of the ACM 41 (2) (1998) 58–65.

[16] P. Bertolazzi, M. Scannapieco, Introducing data quality in a cooperative context, in: Proceedings of the Sixth International Conference on Information Quality (IQ2001), Boston, MA, USA, 2001.

[17] M. Mecella, M. Scannapieco, A. Virgillito, R. Baldoni, T. Catarci, C. Batini, Managing data quality in cooperative information systems, in: Proceedings of the 10th Italian Symposium on Advanced Database Systems (SEBD 2002), Portoferraio (Isola d'Elba), Italy, 2002.

[18] A. Motro, I. Rakov, Estimating the quality of databases, in: T. Andreasen, H. Christiansen, H.L. Larsen (Eds.), Flexible Query Answering Systems, Third International Conference, FQAS'98, Roskilde, Denmark, May 13–15, 1998, Proceedings, Lecture Notes in Computer Science, vol. 1495, Springer, 1998.

[19] F. Naumann, U. Leser, J.C. Freytag, Quality-driven integration of heterogeneous information sources, in: Proceedings of the 25th VLDB, Edinburgh, Scotland, 1999, pp. 447–458.

[20] U. Leser, F. Naumann, Query planning with information quality bounds, in: H.L. Larsen, J. Kacprzyk, S. Zadrozny, T. Andreasen, H. Christiansen (Eds.), Flexible Query Answering Systems, Recent Advances, Proceedings of the Fourth International Conference on Flexible Query Answering Systems, FQAS 2000, Warsaw, Poland, Physica-Verlag, 2000, pp. 85–94.

[21] G.A. Mihaila, L. Raschid, M.-E. Vidal, Using quality of data metadata for source selection and ranking, in: D. Suciu, G. Vossen (Eds.), Proceedings of the Third International Workshop on the Web and Databases, WebDB 2000, Adam's Mark Hotel, Dallas, Texas, USA, May 18–19, 2000, in conjunction with ACM PODS/SIGMOD 2000. Informal proceedings, 2000, pp. 93–98.
[22] T. Lee, S. Bressan, S.E. Madnick, Source attribution for querying against semi-structured documents, in: Workshop on Web Information and Data Management, 1998.
[23] J.I. Maletic, A. Marcus, Data cleansing: beyond integrity checking, in: Proceedings of The Conference on Information Quality (IQ2000), Massachusetts Institute of Technology, October 20–22, 2000, pp. 200–209.
[24] H. Galhardas, D. Florescu, D. Shasha, E. Simon, C.-A. Saita, Improving data cleaning quality using a data lineage facility, in: D. Theodoratos, J. Hammer, M.A. Jeusfeld, M. Staudt (Eds.), Proceedings of the 3rd International Workshop on Design and Management of Data Warehouses, DMDW'2001, Interlaken, Switzerland, June 4, 2001, 2001.
[25] M. Gertz, Managing data quality and integrity in federated databases, in: S. Jajodia, W. List, G.W. McGregor, L. Strous (Eds.), Integrity and Internal Control in Information Systems, IFIP TC11 Working Group 11.5, Second Working Conference on Integrity and Internal Control in Information Systems: Bridging Business Requirements and Research Results, Warrenton, Virginia, USA, November 19–20, 1998, IFIP Conference Proceedings, vol. 136, Kluwer, 1998, pp. 211–230.
[26] D.E. Knuth, The Art of Computer Programming, Fundamental Algorithms, second ed., vol. 1, Addison-Wesley, 1973.
[27] J. Heinsohn, A hybrid approach for modeling uncertainty in terminological logics, Tech. rep., Deutsches Forschungszentrum für Künstliche Intelligenz, 1991.

Martin Peim is a Research Associate in the Information Management Group at the University of Manchester. He obtained his B.Sc. and Ph.D. in mathematics at the University of Manchester and spent two years at the University of Kentucky (1986–1988) before taking up Computer Science. Before joining the IMG in 1999, he worked on distributed decision support systems, automated reasoning and hardware design verification. From 1999 to 2003, he worked on the EPSRC project "Source Integration in Distributed Knowledge-based Query Processing". He is currently working on the Bioquery project, which is a follow-up to the TAMBIS project.

Enrico Franconi has been recently appointed as Associate Professor at the Free University of Bozen-Bolzano, Italy, in the Faculty of Computer Science, where he leads the Knowledge Representation Meets Databases (KRDB) centre. He is currently involved as principal Investigator of the European 5th Framework RTD project "SEmantic Webs and AgentS in Integrated Economies" (SeWAsIE), and as co-investigator of the British Engineering and Physical Sciences Research Council (EPSRC) project "Flexible source integration in distributed knowledge-based query processing for bioinformatic information sources" (TAMBIS-II). He has been Principal Investigator in the European ESPRIT-4 Long Term Research project "Foundations of Data Warehouse Quality" (DWQ). Recently he chaired the 1998 International Description Logics Workshop (DL-98); the 6th International Workshop on Knowledge Representation meets Databases (KRDB-99), associated to IJCAI-99; the PC of the 12th European Summer School in Logic, Language and Information (ESSLLI-2000); the International Workshop on Foundations of Models for Information Integration" (FMII-2001), which is the 10th workshop in the series "Foundations of Models and Languages for Data and Objects" (FMLDO), associated to VLDB-2001; he co-chaired the 9th International Symposium on Temporal Representation and Reasoning (TIME-2002). He has been invited to give keynote talks at various conferences on Conceptual Modelling, Information Integration, and Knowledge Engineering.

Norman W. Paton is a Professor of Computer Science at the University of Manchester. He obtained a B.Sc. in Computing Science from Aberdeen University in 1986, and a Ph.D. from the same institution in 1989. He was a lecturer in Computer Science at Heriot-Watt University, 1989–1995, before moving to Manchester, where he co-leads the Information Management Group. His research interests have principally been in databases, in particular active databases, spatial databases, deductive object-oriented databases and user interfaces to databases. He is currently working on spatio-temporal databases, distributed information systems, Grid data management, and on genome databases.