The Identification of Missing Information Resources by using the Query Difference Operator

Michael Minock, Marek Rusinkiewicz and Brad Perry
MCC
(mjm,marek,bperry)@mcc.com

April 2, 1999

Abstract

In this paper we consider processing of global queries posed over information spaces populated by information resources that may advertise their contents in terms of global domain-specific ontologies. We describe a technique to identify what portions of a user's query may not be answered by a set of available information agents. This is achieved by reasoning over the advertisements of the agents relative to the user's query. The technique to solve this problem is based on the realization that the set difference of the queries q_1 and q_2 may be computed as a syntactic manipulation of the expressions q_1 and q_2 for a well defined subset of the relational algebra over a restricted class of relational schemas. That is to say, one may take the expressions for q_1 and q_2, apply the query difference formula to yield q_3, and be guaranteed that q_3 is logically equivalent to $q_1 - q_2$. With this *Query Difference* operator defined, the ability to compute query intersection, subsumption and equivalence follow. These claims are formally defined and proven and an example from an on-line movie guide domain is provided.

In addition to the identification of missing resource agents, we anticipate a number of other applications of the Query Difference operator. This includes, but is not limited to, limiting the generality of dynamically constructed user queries, efficient query planning, and monitoring user access to sensitive information.

Keywords: Query Difference, Query Failure, Cooperative Query Answering, Intensional Answers.

1 Introduction

The Internet and corporate intranets are currently plagued by a potpourri of keyword-based, low-quality search technologies. End-users are left with the daunting task of formulating complex search expressions to attain manageable results and are left uninformed of the quality or coverage of any result. Efforts in electronic commerce, structured document standards, such as XML, and structured source description standards, such as RDF, have shown the potential to provide conceptual structure to networked information sources with little or no maintenance overhead at the source. As a result, Internet-based access to conceptual data is rapidly becoming a reality. This conceptual structure will enable users to construct focused information requests and data providers to construct accurate content advertisements. Yet, little has been said about how the quality or coverage of a "result" can be computed and conveyed to a user of Internet-style applications. This paper presents the notion of a *query difference* operator that provides a mechanism for computing quality and coverage claims on queries posed over collections of networked information sources.

Conceptual access to networked information sources is accomplished by using dynamic ontologies to describe the expectations of a domain. A *domain ontology* provides a set of concepts over which users may query and data providers may advertise. Autonomous distributed software agents carry out processing tasks and interact using concepts from the domain ontology. For example one agent represents the user, another agent brokers the user request to a set of applicable resources. Resource agents are responsible for a particular data resource or set of resources. Specialized task execution agents carry out more complex tasks, such as active monitoring across sets of resources for complex patterns. To the user, queries and advertisements will have simple natural language equivalents. The InfoSleuth[1] project (Figure 1)[2] at MCC represents one of the first wide scale efforts to deliver this type of capability.

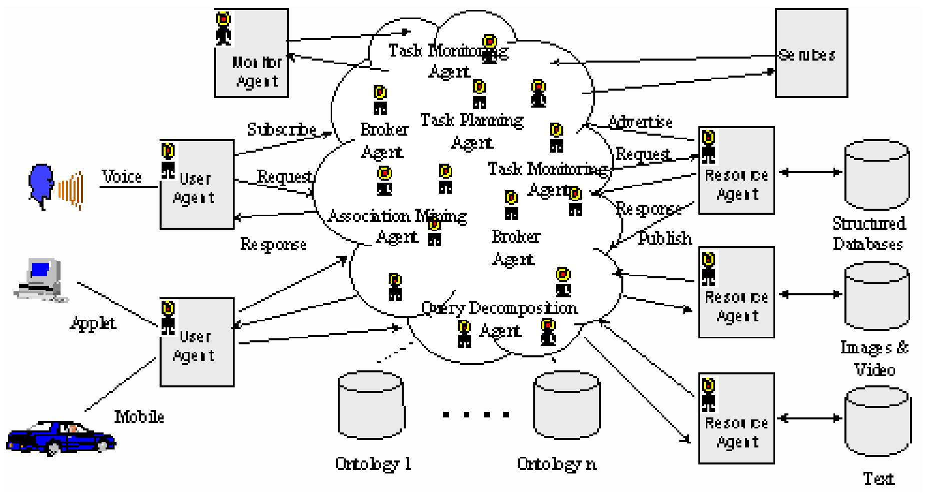

Figure 1: InfoSleuth's Architecture

When a user asks a query in an environment that includes a large number

[1]The InfoSleuth™ Project (http://www.mcc.com/projects/infosleuth) is an R&D project at MCC that is supported by various industrial and government sponsors.

of dynamic, distributed information resources, s/he user will need to be notified when their query, or part of their query is not answerable. For example if a user requests, ‘Give me the list of cardiac pediatricians in Austin, Dallas, or Houston’, then the answer, ‘Smith in Houston, Johnson in Dallas, but there are no resources providing data on doctors in Austin’ is accurate, but, in the same situation, ‘Smith in Houston and Johnson in Dallas’ is misleading. The user may leave with mistaken interpretation that their are no cardiac pediatricians in Austin, while what is actually missing is the information resource agent that provides such information.

The query difference operator introduced in this paper provides distributed information networks, such as InfoSleuth, the ability to compute quality and coverage claims for users’ queries from the “available information” captured in collections of resource agent advertisements.

1.1 Organization of this paper

Section 2 of this paper gives pertinent background work. Section 3 presents and proves the correctness of the query difference formula and discusses techniques to simplify query expressions. Sections 4 compares this work to prior work and proposes some future directions and additional applications of the query difference operator.

2 Background

Internet-style information applications are characterized by:

- Very large numbers of information sources (potentially thousands or more).
- No formal model of control, administration, or registration of information sources.
- Dynamic availability of sources and evolving content available from any one source.

Such systems are in contrast to federated database systems, where a global schema is decided a priori, but no additional databases are allowed to join (unless their schema exactly match the global schema and the centralized system is reconfigured). These systems are also beyond heterogeneous databases, where a merge of the schemas of the participating databases becomes the global schema. These systems are also beyond data warehousing technologies which, at fixed times and over fixed resources, simply aggregate data in a central repository. In agent based distributed informations systems it is an ontology model that is primary. All queries and all information sources are mapped as views on this domain model. Distributed software agents, sharing the ontology as common knowledge, give the overall system greater degrees of fault-tolerance and dynamism than previous approaches.

However before such systems may be widely used, a set of cooperative techniques must be employed to avoid misleading interpretation of system answers. For example, similar to asking a question to multiple people with unknown quality of belief, when users asks a query over multiple data resource agents they would like to know where the agents agree, disagree, or do not know. If many agents agree on a particular data value, and the user is made aware of this agreement, the user will have more confidence in the result. If there are differences in belief, then the user may decide which value to believe based on their judgment of the reliability of the various agents. The straight forward approach to the problem seems to suffice. That is simply mark each answer tuple with the source from which it came. Let the user quickly browse data by its source and provide tools that allow for quick cross-referencing of resource answers. *The problem we focus on here is what happens if no available agent can answer the query (or, more importantly, part of the query).* The user would like to know which additional agents, or what additional information, would be required to fully answer their query.

The technique to solve this problem is based on the realization that the set difference of the queries q_1 and q_2 may be computed as a syntactic manipulation of the expressions q_1 and q_2 for a well defined subset of the relational algebra over a restricted class of relational schemas. That is to say, one may take the expressions for q_1 and q_2, apply the query difference formula to yield q_3, and be guaranteed that q_3 is logically $q_1 - q_2$. With query set difference handled, the ability to compute query intersection, subsumption and equivalence follow.

As an example consider q_1 being the query *"Give the show times in Austin for G or PG-13 movies with 4 or 5 star evaluations"* and q_2 being the query *"Give the show times in Austin or San Antonio for PG-13 or R movies with 3 or 4 star evaluations."* The logical result of $q_1 - q_2$ is the query *"Give show times in Austin for G movies with 4 or 5 star evaluations plus show times in Austin for PG-13 movies with 5 star evaluations."* Similarly, $q_1 \cap q_2$ is *"Give show times in Austin for PG-13 movies with 4 star evaluations."* Clearly neither query subsumes the other, nor are they equivalent. Note that if q_1 was the user's query, and q_2 was what the system knew, then $q_1 - q_2$ is the part of the user's query that is not answerable or not covered by the information known to the system.

2.1 Application Example

The example that follows pertains directly to the InfoSleuth system at MCC, but provides an "application framework" applicable to a wide class of distributed information systems. In InfoSleuth[2] a user, through their user agent, issues a query relative to a common domain ontology. This query is passed to a query

[2]The definition of InfoSleuth here is a gross simplification of the actual InfoSleuth system. However, although it leaves out discussion of ontology creation and management, image, free text, web-page, and object data, complex event detection, the intricacies of agent engineering and value mapping, it does capture enough of the spirit of the system for discussion here. We refer the reader to [2][7] for an in depth discussion of InfoSleuth.

agent, that, after consulting with a broker agent that possesses resource agent advertisements, decides the set relevant resource agents, sends these resource agents parts of the query, waits on results from these resources, fuses these results into an answer to the original query that is passed back to the user agent.

Consider the domain ontology (or schema) for a simple online movie, review , and show-time system [3].

```
Movie(title, year, type)
Show(title, theater, time, city)
Review(title, source, eval)
```

Now consider that we have resource agents advertise what information they are able to provide over this ontology. For example one agent declares that it knows all about action movies made between 1970 and 1998. Another resource agent states that it has all the reviews ever made by Gene Siskel. Still another resource advertises that it is aware of the movie showings in Austin. A resource may include conditions in its advertisements external to the actual data it holds. For instance an agent may claim to have all the show times in Austin for the Movies that Ebert reviewed in the last ten years.

It is easy to envision many thousands of resource agents distributed over the Internet, dynamically coming and going, each claiming a coverage over part of this ontology.

An assumption that underlies all the algorithms in this paper is that domain ontology may be expressed as a unique *Universal Relation*[15][16][6][11]. This is true for the example movie schema and this Universal Relation is:

title	movie		show			review	
title	year	type	theater	time	city	source	eval

3 Query Analysis

3.1 Definitions and Assumptions

Assume a set of relations R where $R = R_1, ..., R_m$ and R may be expressed as a Universal Relation.

In our example the sub-relations are $\{title, movie, show, review\}$.

A query is represented as a relational algebra expression of the form $\Pi_X \sigma_{c_1 \wedge \ldots \wedge c_n}$ where X identifies a set of sub-relations that make up R. The conditions $c_1, ..., c_n$ are simple, non-join conditions. Note that the universal relation R is the "relation" that these queries are applied over. Because this is always the case, "R" is omitted from the notation. So the query retrieving all the show times for recent horror films playing in Austin is:

$\Pi_{title,show} \sigma_{movie.year>1980 \wedge movie.type=horror \wedge show.city=Austin}$

[3]This example is based on the example used in [8].

title	movie		show			review	
title	year	type	theater	time	city	source	eval
Star Wars	77	SciFi	Gateway	10 pm	Austin	NULL	NULL
Rtrn Jedi	81	SciFi	NULL	NULL	NULL	Siskel	up
...	...	...	...	...	...	...	...

Table 1: An instance of the universal relation.

The tuple τ is a row in the Universal Relation, padded, if necessary, with possible null values. The example Universal Relation in Table 1 shows 2 such tuples.

The tuple τ may be (a part of) an answer to a query, in which case we shall write $\Pi_X \tau \in \Pi_X \sigma_{C_1 \wedge \ldots \wedge C_n}$. For example assume that τ_1 is the *Star Wars* tuple in table 1, then $\Pi_{title,movie}\tau_1 \in \Pi_{title,movie}\sigma_{city=Austin}$. For the time being we shall adopt a closed world assumption. That is if a tuple has a NULL value for an attribute, then a simple condition on that attribute will evaluate to false. For example $\Pi_{title,movie}\tau_1 \notin \Pi_{title,movie}\sigma_{eval=fivestar}$.

As a basic definition of extended[4] projections over tuples, we have:
$\Pi_X \tau \in \Pi_X \sigma_{c_1 \wedge \ldots \wedge c_n} \Rightarrow \Pi_Y \tau \in \Pi_Y \sigma_{c_1 \wedge \ldots \wedge c_n}$, if $Y \subseteq X$
and:
$\Pi_X \tau \in \Pi_X \sigma_{c_1 \wedge \ldots \wedge c_n} \Rightarrow \Pi_Y \tau \in \Pi_X \sigma_{c_1 \wedge \ldots \wedge c_n}$, if $Y \subseteq X$
and:
$\Pi_X \tau \in \Pi_X \sigma_{c_1 \wedge \ldots \wedge c_n} \Rightarrow \Pi_X \tau \in \Pi_X \sigma_C$, if $C \subseteq \{c_1, ..., c_n\}$.

So for example, if $\Pi_{title,movie,show}\tau_1 \in \Pi_{title,movie,show}\sigma_{year=77 \wedge type=SciFi \wedge city=Austin}$, then we also know that $\Pi_{title}\tau_1 \in \Pi_{title,show}\sigma_{\wedge type=SciFi \wedge city=Austin}$.

The set difference and set union operators of the relational algebra are applied to "union-compatible" relations - i.e. $R - S$ or $R \cup S$ only make sense if R and S have equivalent sets of attributes. Here these notions are generalized to enable the union and difference over non "union-compatible" relations. The corresponding operations are super-imposition $\oplus$ and difference $\ominus$.

Definition 1 *(Super-imposition)* $\Pi_j \tau \in \Pi_X \sigma_{c_1 \wedge \ldots \wedge c_n} \oplus \Pi_Y \sigma_{c'_1 \wedge \ldots \wedge c'_{n'}}$ *iff*
$\Pi_j \tau \in \Pi_{X \cap Y} \sigma_{c_1 \wedge \ldots \wedge c_n} \vee \Pi_j \tau \in \Pi_{X \cap Y} \sigma_{c'_1 \wedge \ldots \wedge c'_{n'}}$ *for* $j \in X \cap Y$
$\Pi_j \tau \in \Pi_{X-Y} \sigma_{c_1 \wedge \ldots \wedge c_n}$ *for* $j \in X - Y$
$\Pi_j \tau \in \Pi_{Y-X} \sigma_{c'_1 \wedge \ldots \wedge c'_{n'}}$ *for* $j \in Y - X$

Definition 2 *(Difference)* $\Pi_j \tau \in \Pi_X \sigma_{c_1 \wedge \ldots \wedge c_n} \ominus \Pi_Y \sigma_{c'_1 \wedge \ldots \wedge c'_{n'}}$ *iff*
$\Pi_j \tau \in \Pi_X \sigma_{c_1 \wedge \ldots \wedge c_n} \wedge \Pi_j \tau \notin \Pi_Y \sigma_{c'_1 \wedge \ldots \wedge c'_{n'}}$ *for* $j \in X \wedge j \in Y$
$\Pi_j \tau \in \Pi_X \sigma_{c_1 \wedge \ldots \wedge c_n}$ *for* $j \in X \wedge j \notin Y$

[4]Note that under this definition of $\Pi_X \tau \in \Pi_Y \sigma_C$ the projection X and Y are non necessarily "union-compatible".

title	movie		show			review	
title	year	type	theater	time	city	source	eval
Star Wars	77	SciFi					
Rtrn Jedi	81	SciFi					
...	...	...					

Table 2: $\Pi_{title,movie}\sigma_{type=SciFi}$

title	movie		show			review	
title	year	type	theater	time	city	source	eval
Star Wars			Gateway	10 pm	Austin		
...			...	...	...		

Table 3: $\Pi_{title,show}\sigma_{city=Austin}$

title	movie		show			review	
title	year	type	theater	time	city	source	eval
Star Wars	77	SciFi	Gateway	10 pm	Austin		
Rtrn Jedi	81	SciFi					
...	...	...	...	...	...		

Table 4: $\Pi_{title,movie}\sigma_{type=SciFi} \oplus \Pi_{title,show}\sigma_{city=Austin}$

title	movie		show			review	
title	year	type	theater	time	city	source	eval
	77	SciFi					
Rtrn Jedi	81	SciFi					
..	..	..					

Table 5: $\Pi_{title,movie}\sigma_{type=SciFi} \ominus \Pi_{title,show}\sigma_{city=Austin}$

Tables 2 and 3 show the results of 2 simple queries over the Universal Relation in Table 1. Table 4 shows the super-imposition of these queries. Table 5 shows the difference of these queries.

3.2 The Query Difference Operator

Here follows the main result of this paper. This theorem says that for the simple relational algebra queries of the type proposed, one may solve for query difference by manipulating query expressions according to the following formula.

Theorem 1 *(Query Difference Formula)*
$\Pi_X \sigma_{c_1 \wedge \ldots \wedge c_n} \ominus \Pi_Y \sigma_{c'_1 \wedge \ldots \wedge c'_{n'}} =$
$\Pi_{X \cap Y} \sigma_{c_1 \wedge \ldots \wedge c_n \wedge \neg c'_1} \oplus \ldots \oplus \Pi_{X \cap Y} \sigma_{c_1 \wedge \ldots \wedge c_n \wedge \neg c'_{n'}} \oplus$
$\Pi_{X-Y} \sigma_{c_1 \wedge \ldots \wedge c_n}$

Proof:

First we shall prove soundness: That is if $\Pi_j \tau \in \Pi_{X \cap Y} \sigma_{c_1 \wedge \ldots \wedge c_n \wedge \neg c'_1} \oplus \ldots \oplus \Pi_{X \cap Y} \sigma_{c_1 \wedge \ldots \wedge c_n \wedge \neg c'_{n'}} \oplus \Pi_{X-Y} \sigma_{c_1 \wedge \ldots \wedge c_n}$ then $\Pi_j \tau \in \Pi_X \sigma_{c_1 \wedge \ldots \wedge c_n}$ and $\Pi_j \tau \notin \Pi_Y \sigma_{c'_1 \wedge \ldots \wedge c'_{n'}}$ for $j \in X \wedge j \in Y$ and $\Pi_j \tau \in \Pi_X \sigma_{c_1 \wedge \ldots \wedge c_n}$ for $j \in X \wedge j \notin Y$.

We shall conduct the proof term by term.

Assume that $\Pi_j \tau \in \Pi_{X \cap Y} \sigma_{c_1 \wedge \ldots \wedge c_n \wedge \neg c'_i}$ for an arbitrary i - in such a case we know $j \in X \wedge j \in Y$. Let us now assume that $\Pi_j \tau \in \Pi_Y \sigma_{c'_1 \wedge \ldots \wedge c'_{n'}}$. This would mean that $\Pi_j \tau \in \Pi_{Y \cap X} \sigma_{c'_1 \wedge \ldots \wedge c'_{n'}}$. But this violates our assumption, $\Pi_j \tau \in \Pi_{X \cap Y} \sigma_{c_1 \wedge \ldots \wedge c_n \wedge \neg c'_i}$. Hence, $\Pi_j \tau \notin \Pi_Y \sigma_{c'_1 \wedge \ldots \wedge c'_{n'}}$. It is also clear that $\Pi_j \tau \in \Pi_{X \cap Y} \sigma_{c_1 \wedge \ldots \wedge c_n}$ this finally establishes that $\Pi_j \tau \in \Pi_X \sigma_{c_1 \wedge \ldots \wedge c_n}$ for $j \in X$ and $j \in Y$.

Assume that $\Pi_j \tau \in \Pi_{X-Y} \sigma_{c_1 \wedge \ldots \wedge c_n}$. Clearly $j \notin Y$ and $j \in X$. Hence we may write $\Pi_j \tau \in \Pi_X \sigma_{c_1 \wedge \ldots \wedge c_n}$ for $j \in X \wedge j \notin Y$.

Taking both together we have shown soundness of the difference formula.

Now we shall prove completeness: That is if $\Pi_j \tau \in \Pi_X \sigma_{c_1 \wedge \ldots \wedge c_n}$ and $\Pi_j \tau \notin \Pi_Y \sigma_{c'_1 \wedge \ldots \wedge c'_{n'}}$ for $j \in X \wedge j \in Y$ or if $\Pi_j \tau \in \Pi_X \sigma_{c_1 \wedge \ldots \wedge c_n}$ for $j \in X \wedge j \notin Y$ then $\Pi_j \tau \in \Pi_{X \cap Y} \sigma_{c_1 \wedge \ldots \wedge c_n \wedge \neg c'_1} \oplus \ldots \oplus \Pi_{X \cap Y} \sigma_{c_1 \wedge \ldots \wedge c_n \wedge \neg c'_{n'}} \oplus \Pi_{X-Y} \sigma_{c_1 \wedge \ldots \wedge c_n}$

The proof shall be by the two cases: If $\Pi_j \tau \in \Pi_X \sigma_{c_1 \wedge \ldots \wedge c_n}$ and $\Pi_j \tau \notin \Pi_Y \sigma_{c'_1 \wedge \ldots \wedge c'_{n'}}$ for $j \in X \wedge j \in Y$ then for some c'_i, $\Pi_j \tau \in \Pi_{X \cap Y} \sigma_{c_1 \wedge \ldots \wedge c_n \wedge \neg c'_i}$, because $\neg \exists c'_i \Pi_j \tau \in \Pi_{X \cap Y} \sigma_{c_1 \wedge \ldots \wedge c_n \wedge \neg c'_i} \Rightarrow \Pi_j \tau \in \Pi_{X \cap Y}\ \sigma_{c_1 \wedge \ldots \wedge c_n \wedge c'_1 \wedge \ldots \wedge c'_{n'}} \Rightarrow \tau \in \Pi_{X \cap Y} \sigma_{c'_1 \wedge \ldots \wedge c'_{n'}}$ which violates the assumption $\Pi_j \tau \notin \Pi_Y \sigma_{c'_1 \wedge \ldots \wedge c'_{n'}}$ for $j \in X \wedge j \in Y$. Hence all cases where $j \in Y$ will be covered by a term on the right-hand side of the difference formula.

If $\Pi_j \tau \in \Pi_X \sigma_{c_1 \wedge \ldots \wedge c_n}$ and $j \notin Y$, then it is trivially true that $\Pi_j \tau \in \Pi_{X-Y} \sigma_{c_1 \wedge \ldots \wedge c_n}$. Hence all cases where $j \notin Y$ will be covered by a term on the right-hand side of the difference formula.

Thus it has been shown that the query difference is both sound and complete. •

Take our simple example from before. Following the query difference theorem we compute $\Pi_{title,movie}\sigma_{type=SciFi} \ominus \Pi_{title,show}\sigma_{city=Austin}$ equal to $\Pi_{title}\sigma_{type=SciFi \wedge \neg city=Austin} \oplus \Pi_{movie}\sigma_{type=SciFi}$. If you apply this query to the Universal Relation in Table 2 you obtain the results in Table 5.

The following theorem generalizes the results of the previous theorem to include compound queries. That is queries consisting of a set of the simple relational algebra queries super-imposed.

Theorem 2 *(Query Difference is distributive over compound queries)*
$(q_1 \oplus q_2) \ominus (q_3 \oplus q_4) = ((q_1 \ominus q_3) \ominus q_4) \oplus ((q_2 \ominus q_3) \ominus q_4)$

Proof:
omitted for the sake of brevity. •

Because $\ominus$ is sound and complete one may reliably compute query differences. In addition one may use this operation to compute subsumption and intersection between query expressions. Subsumption is easy. If query $q_1 \ominus q_2 = \emptyset$ then q_2 subsumes q_1. Subsumption is important for determining if the query is fully covered by resource agent advertisements. Intersection is also easy $q_1 \cap q_2 = q_1 \ominus (q_1 \ominus q_2)$. Intersection may be used to decide minimal queries with respect to resource agents. Finally q_1 and q_2 are equivalent iff $q_1 \ominus q_2 = \emptyset$ and $q_2 \ominus q_1 = \emptyset$.

3.3 Query Simplification

In the worst case, the subtraction of k simple queries, each with exactly n conditions, from an initial simple query may yield a compound query expression with n^k terms. Although it should be noted that this worst case is only realized when the nk simple conditions are all on different attributes. In practice many terms are unsatisfiable and simply cancel out.

However because of dire worst case query lengths[5], it is necessary to apply as many simplifications to the query expression as possible. We shall now assert the set of simplification rules that will allow us to reduce the number of terms in a query expression.

Theorem 3 *(Absorption)* $\Pi_X\sigma_C \oplus \Pi_Y\sigma_{C'} = \Pi_Y\sigma_{C'}$
if $\Pi_Y\tau \in \Pi_Y\sigma_C \Rightarrow \Pi_Y\tau \in \Pi_Y\sigma_{C'}$ *and* $X \subseteq Y$.

$\Pi_{title,show}\sigma_{city=Austin \wedge time<9pm} \oplus \Pi_{title,movie,show}\sigma_{city=Austin} \equiv$
$\Pi_{title,movie,show}\sigma_{city=Austin}$.

Theorem 4 *(Horizontal Merge)* $\Pi_X\sigma_C \oplus \Pi_Y\sigma_C = \Pi_{X\cup Y}\sigma_C$

$\Pi_{title,show}\sigma_{city=Austin} \oplus \Pi_{title,review}\sigma_{city=Austin} \equiv$
$\Pi_{title,show,review}\sigma_{city=Austin}$.

[5]Natural language descriptions of the queries might also be presented to users. In presenting such descriptions it would be advantageous to minimize the number of query terms[14].

Theorem 5 *(Vertical Merge)* $\Pi_X \sigma_{C \wedge c_i} \oplus \Pi_X \sigma_{C \wedge c_j} = \Pi_X \sigma_{C \wedge c_k}$ *where* $c_i \vee c_j$ *may be written as the simple condition* c_k.

$\Pi_{title,show} \sigma_{city=Austin \wedge time<9pm \wedge theater=Dobie} \oplus$
$\Pi_{title,show} \sigma_{city=Austin \wedge time<9pm \wedge theater=Gateway} \equiv$
$\Pi_{title,show} \sigma_{city=Austin \wedge time<9pm \wedge theater=Gateway \vee Dobie}$.

Proofs:
omitted for the sake of brevity. •

We conjecture that these three axioms are complete with respect to simplifying compound query expressions.

3.4 Limitations

There are some caveats[6] to the blind use of the Query Difference Operator. One must be careful when simplifying the terms in resulting query expressions. Consider computing $\Pi_{people} \sigma_{work=MCC} \ominus \Pi_{people} \sigma_{work \neq MCC}$. The query difference operator yields the result $\Pi_{people} \sigma_{work=MCC \wedge \neg(work \neq MCC)}$. Note that because a person may work for more than one company, this result is not equal to simply $\Pi_{people} \sigma_{work=MCC}$. In fact the expression $work = MCC \wedge \neg(work \neq MCC)$ is long hand way of specifying universal quantification, meaning that matching tuples work for *MCC only*.

The strategy to adopt in these simplifications is to only allow the negation to distribute over the condition when the attribute that is the subject of the condition is functionally determined by the key (if it exists) of the projection set of the query. In the final analysis, when relational algebra queries are translated to SQL, embedded queries may need to enforce universal quantification.

3.5 Example

The above formulas lead to the following process for computing query coverage and the identification of missing resources in a network of information sources:

- `Advertisements`: Let $A = \{q_1, q_2, \ldots q_n\}$ be the set of advertisements from resource agents. That is, each resource agent's advertisement is one or more queries expressing its content coverage.
- `User Query`: Let q_u represent a user's query to the system.
- `Query Analysis`: Let $q_x = q_u \ominus A = (q_u \ominus q_1) \oplus \ldots \oplus (q_u \ominus q_n)$.
- `Query Coverage`: The expression q_x is the part of the user's query that is unanswerable, or incomplete, from the current set of resources available to the system.

[6] This section is expected to grow as we experiment more with the system and get feedback from the community.

TextBook
$\Pi_{title,movie,review}$ $\sigma_{movie.type \in \{drama,comedy\} \wedge movie.year<1983 \wedge movie.year>1927}$ “Titles, movie information, and reviews for all the drama and comedy films made between 1927 and 1983.”

Catalog
$\Pi_{title,movie}$ $\sigma_{movie.type \in \{action,documentary,drama,comedy\} \wedge movie.year<1999 \wedge movie.year>1955}$ “Titles and movie information for all the action, documentary, drama, and comedy movies made between 1955 and 1999.”

VOID
$\Pi_{title,movie,review,show}\sigma_{movie.year<=1927} \oplus$ $\Pi_{title,movie,review,show}\sigma_{movie.year>=1999}$ “All films have been made after 1927 and before 1999”

Table 6: Resource agent advertisements in simple example.

In this section we shall demonstrate the results of this system as implemented in a simple prototype. We have tested some very large, complex examples in this system with reasonable results and performance. The example here, however, is for illustration purposes.

Table 6 shows the advertisements for the example. There are 2 real resources here and 1 virtual resource which is used to enforce an integrity constraint[7]. Tables 7,8, and 9 show the results obtained for three different queries. Table 7 shows a complete query plan for the user's query, whereas Tables 8 and 9 only show the explanation of what additional information is needed.

4 Comparison to Previous Work and Future Directions

There has been a great deal of work dealing with the optimization of relational algebra expressions. To our knowledge none of this work has used a DeMorgan type transforms to directly solve query difference at a syntactic level. This is in large part because we have made a Universal Relation assumption whereas work in traditional query optimization has assumed that the schema is in a less restrictive form.

[7]Our apologies to silent film fans.

USER QUERY
$\Pi_{title,movie}\sigma_{movie.year<1960 \wedge movie.year>=1930}$ "Give me all the titles and movie information for films made in the thirties, forties, or fifties."
Query to *TextBook*
$\Pi_{title,movie}\sigma_{movie.type\in\{drama,comedy\} \wedge movie.year<1960 \wedge movie.year>=1930}$
Query to *Catalog*
$\Pi_{title,movie}\sigma_{movie.type\in\{action,documentary,drama,comedy\} \wedge movie.year<1960 \wedge movie.year>1955}$
Explanation of Incompleteness
$\Pi_{title,movie}\sigma_{movie.year<1960 \wedge movie.year>=1930 \wedge \notin\{action,documentary,drama,comedy\}} \oplus$ $\Pi_{title,movie}\sigma_{movie.year<1955 \wedge movie.year>=1930 \wedge \notin\{drama,comedy\}}$ "No title and movie information for non drama or comedy films made between 1930 and 1955. Also no title and movie information for non action, drama, comedy, or documentary films made between 1930 and 1960."

Table 7: Full plan and explanation of missing data for first query

USER QUERY
$\Pi_{title,movie,review}\sigma_{movie.type\in\{drama\}}$ "Give me titles, movie information, and reviews for all dramas."
Explanation of Incompleteness
$\Pi_{review}\sigma_{movie.type\in\{drama\} \wedge movie.year<1999 \wedge movie.year>=1983}$ "There is no review information for dramas made after 1982."

Table 8: Explanation of missing data for second query

USER QUERY
$\Pi_{title,movie,review}\sigma_{movie.type \in \{drama,documentary\} \wedge movie.year>=1950 \wedge movie.year<1960}$ “Give me all the titles, movie information, and reviews for all the dramas and documentaries made in the 1950’s.”
Explanation of Incompleteness $\Pi_{title,movie,review}\sigma_{movie.type \in \{documentary\} \wedge movie.year>=1950 \wedge movie.year<1955} \oplus$ $\Pi_{review}\sigma_{movie.type \in \{documentary\} \wedge movie.year>=1955 \wedge movie.year<1960}$ “There in no title, movie information, or review for documentaries made between 1950 and 1955. There is no review information for documentaries made between 1955 and 1960.”

Table 9: Explanation of missing data for third query

Work in description logics[5] has relied heavily on determining query containment. The heart of these systems consist of structural comparison algorithms which, after converting concept expressions to a normal form, determine, on a piece by piece basis, whether one concept subsumes another. However, because of interactions between sub-patterns, these algorithms are generally incomplete.

The notions of determining validity and completeness in relational databases has been explored in [13] and [4]. The question of whether a query is valid (contains only correct information) or complete (contains all information) is computed through a process of rewriting the user’s query over predefined valid or complete views.

Recently[8] considered the problem of obtaining complete answers from incomplete databases. This problem is addressed by viewing it as a case of the problem of independence of queries from updates. By doing so, the problem of determining if a query answer is complete may be computed by relying on a prior technique (see [9]). Although the issue of whether a given query may be completely covered by a set of complete databases is addressed, the issue of explaining that portion of the query that is responsible for the incompleteness is not addressed. In the context of this paper this translates into determining whether a query is answerable, in its totality, over a set of advertised, complete databases.

The SIMS project[1] has addressed the issue of incomplete queries by leveraging LOOM’s [10] classifier. Again, this is based on the notion of query containment, and aside from identifying the class that is missing information, there is no capability to explain *exactly* what portion of the query is responsible for the incompleteness.

There has been work on the notion of residues in the context of Semantic Query Optimization in deductive databases [3]. A residue is the “interaction” of an integrity constraint and an intensional axiom. Such residues are used to speed up query processing.

4.1 Future Applications and Research

Presuming that a common knowledge ontology may be converted to a universal relation, and assuming that resource agent advertisements and user queries are either simple or compound queries of the relational algebra form defined above, several immediate practical possibilities follow.

The queries sent to resources are logically the intersection of the user query with what the resource agent claims to know. This means that queries sent to resources are logically minimal. Hence query difference may be useful to plan queries where the result is an *outer join* of the queries sent to all of the resources.

In all information systems there is the perennial danger that the user will ask a very broad query, which will return a very large number of answers. For example, "Give me all show information (for all theaters in the U.S.)". Giving novice users the ability to compose arbitrary queries, such as in TQML[12] formulator, accentuates this problem. We propose here that a system administrator define a set of *Maximally General Queries (MGQ).* The user would only be allowed to ask queries that are subsumed by a query in the MGQ set. If the user's query is not subsumed by any queries in the MGQ, the user would be asked to be more precise, and might even be assisted by being given the set of MGQ query intersections with their query.

The ability to compute query intersections may be used to determine if users are accessing sensitive information. For example a parent may wish to insert the pattern $\Pi_{title,movie,show,review}\sigma_{movie.type=adult}$ into the system and be informed when their child retrieves *any* information intersecting with this pattern.

If the keys from one relation are contained in the other, then we are able to propagate the constraints of the former onto the initial. Through this technique we may make advertisements more precise.

The query difference operator may also be used to identify resource agent overlaps in data, and hence be the handle on cross checking data belief among a set of resource agents.

This paper has assumed agents are making closed world claims over the domain ontology. In the future we shall explore different assumptions. An advertisement might be an *open world* claim. It might be accompanied by a certainty measure. In the final analysis there may be a set of different types of claims an agent can make. Building an architecture that could accommodate these types of claims, supplemented with the Query Difference Operator may move us closer to the reality of distributed cooperative information systems.

5 Conclusions

This paper has given a solution to the problem identifying the exact portion of a user's query that is not answerable over a set of resources agent advertisements. The *Query Difference Operator* is used at the heart of the algorithm that makes this calculation. Though the query difference operator may only be used for a subset of the relational algebra over the set of schemas that may be expressed as a Universal Relation, we believe that this covers a set of real world problems. Through using the query difference operator, the relational algebra becomes a "natural" algebra - one may solve query equations for unknowns. We anticipate that work over the coming period will further clarify exactly what systems may be treated with the query difference operators, and what techniques may be employed to approximate more complex systems so that the query difference operator may be applied to these systems as well.

6 Bibliography

References

[1] Yigal Arens, Chung-Nan Hsu, and Craig A. Knoblock. Query processing in the SIMS information mediator. In Austin Tate, editor, *Advanced Planning Technology*, pages 61–69. AAAI Press, Menlo Park, California, 1996.

[2] R. J. Bayardo Jr, W. Bohrer, R. Brice, A. Cichocki, J. Fowler, A. Helal, V. Kashyap, T. Ksiezyk, G. Martin, M. Nodine, M. Rashid, M. Rusinkiewicz, R. Shea, C. Unnikrishnan, A. Unruh, and D. Woelk. The InfoSleuth project. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 26(2):543–??, ???? 1997.

[3] U. S. Chakravarthy, J. Grant, and J. Minker. Foundations of semantic query optimization for deductive databases. In J. Minker, editor, *Proc. Workshop on Foundations of Deductive Databases and Logic Programming*, pages 67–101, Washington, D.C., August 18-22, 1986.

[4] R. Demolombe. Validity queries and completeness queries. *Lecture Notes in Computer Science*, 1079:253–??, 1996.

[5] F.-M. Donini, M. Lenzerini, D. Nardi, and A. Schaerf. Reasoning in description logics. In G. Brewka, editor, *Principles of Knowledge Representation*, pages 191–236. CSLI Publications, Stanford (CA), USA, 1996.

[6] R. Fagin and J.D. Ullman A. Mendelzon. A simplified universal relation assumption and its properties. Rj2900, IBM Research Labs, San jose, California, 1980.

[7] Jerry Fowler, Marian Nodine, Brad Perry, and Bruce Bargmeyer. Agent-based semantic interoperability in infosleuth. *Sigmod Record*, 28, 1999.

[8] A. Levy. Obtaining complete answers from incomplete databases. In T. M. Vijayaraman et al., editors, *Proceedings of the twenty-second international Conference on Very Large Data Bases, September 3–6, 1996, Mumbai (Bombay), India*, pages 402–412, Los Altos, CA 94022, USA, 1996. Morgan Kaufmann Publishers.

[9] A. Y. Levy and Y. Sagiv. Queries independent of updates. In Rakesh Agrawal, Sean Baker, and David Bell, editors, *Very large data bases, VLDB '93: proceedings of the 19th International Conference on Very Large Data Bases, August 24–27, 1993, Dublin, Ireland*, pages 171–181, Palo Alto, Calif., USA, 1993. Morgan Kaufmann Publishers.

[10] R. MacGregor and M. Burstein. Using a descriptive classifier to enhance knowledge representation. *IEEE Expert*, 6(3):41–47, 1991.

[11] D. Maier and J. Ullman. Maximal objects and the semantics of universal relation database s. *ACM Transactions on Database Systems*, 8(1):1–14, March 1983.

[12] M. Minock and J. Fowler. Template query mark-up language (tqml) 1.0. Technical Report INSL-122-98(P), MCC, 1998.

[13] Amihai Motro. Integrity = validity + completeness. *ACM Transactions on Database Systems*, 14(4):480–502, December 1989.

[14] C. Shum and R. Muntz. Implicit representation for extensional answers. In L. Hershberg, editor, *Expert Database Systems*. Tysons Corner, 1987.

[15] J.D. Ullman. *Principles of Database Systems*. Computer Science Press, 1982.

[16] J.D. Ullman. *Principles of Database and Knowledge-Base Systems (Seco nd Edition)*. Computer Science Press, Inc., Rockville, Maryland, 1989.