Semantic Queries in Databases: Problems and Challenges

Lipyeow Lim*
University of Hawaii at Manoa
Honolulu, HI 96822, USA
lipyeow@hawaii.edu

Haixun Wang†
Microsoft Research Asia
Beijing China 100190
haixunw@microsoft.com

Min Wang
IBM T. J. Watson Research Ctr
Hawthorne, NY 10532, USA
min@us.ibm.com

ABSTRACT

Supporting semantic queries in relational databases is essential to many advanced applications. Recently, with the increasing use of ontology in various applications, the need for querying relational data together with its related ontology has become more urgent. In this paper, we identify and discuss the problem of querying relational data with its ontologies. Two fundamental challenges make the problem interesting. First, it is extremely difficult to express queries against graph structured ontology in the relational query language SQL, and second, in many cases where data and its related ontology are complicated, queries are usually not precise, that is, users often have only a vague notion, rather than a clear understanding and definition, of what they query for. We outline a query-by-example approach that enables us to support semantic queries in relational databases with ease. Instead of endeavoring to incorporate ontology into relational form and create new language constructs to express such queries, we ask the user to provide a small number of examples that satisfy the query she has in mind. Using these examples as seeds, the system infers the exact query automatically, and the user is therefore shielded from the complexity of interfacing with the ontology.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Query formulation*; H.5 [**Information Interfaces and Presentation**]: Miscellaneous

General Terms

Algorithms

1. INTRODUCTION

There is an urgent need to incorporate ontology into the realm of object relational databases (ORDBMs) so that the user can query data and its related ontology in a consistent manner [6, 12, 18]. Fueling this need to support semantic queries in relational databases is a growing number of advanced applications such as product information management (PIM) systems, customer relationship management (CRM) systems, electronic medical records (EMRs) systems, etc.

Consider an EMR (electronic medical records) system. Clinicians recording the diagnosis of a patient visit may choose different disease codes for the same symptoms that the patient is exhibiting. One clinician might describe a patient diagnosis using the code for "Tumor of the Uvea", while another might use the code for "Iris Neoplasm". In the patient's EMR a generic term like "Eye Neoplasm" might be recorded instead of the more specific "Tumor of the Uvea" (we will use the more descriptive terms instead of the corresponding codes in this paper in the interest of readability). Hence, in order to obtain meaningful results from querying an EMR database, the query processing system needs to understand the semantics of the query and the data.

The need for data semantics has spurred the increasing use of ontology in various applications. Continuing with the EMR example, many ontologies have been developed to capture the semantics of various sub-components of EMR data. For example, the National Cancer Institute (NCI) Thesaurus [14] is a collection of ontologies spanning the following areas: Drugs and Chemicals, Diseases Disorders and Findings, Anatomic Structure System or Substance, Gene, Chemotherapy Regimen etc. Fig. 2 shows a fragment of the NCI Thesaurus in graphical form. Moreover, many of these ontologies are well integrated with existing data coding terminologies (eg, SNOMED [9], ICD9 [8]) used by industry EMR formats such as HL7 [7] and CCR [5]. With the confluence of ontologies, coding terminologies, and data standards, the need for querying relational data along with its related ontology has become even more urgent.

Although we recognize the importance of querying relational data along with its related ontology, it is extremely tedious and time consuming to understand ontology, and use ontology in database queries. The success of the relational database technology is at least partly due to the spartan simplicity of its data model and query language, which insulate the user from the physical implementation of the database. But for semantic queries, users are often exposed to the full complexity of the ontology. Still, integrating data and its related ontology is a challenge too important to ignore. There are two major approaches in attacking this problem. One is to flatten graph-structured ontologies into relational form [6, 18], and the other is to extend ORDBMSs and SQL to handle non-relational data directly [11, 12]. However, both approaches incur tremendous system cost, but have limited success in taking the tediousness out of handling semantic queries.

In this paper, we discuss the problems and challenges of support-

*Work done while author is at IBM T J Watson.
†Work done while author is at IBM T J Watson.

CIKM'09, November 2–6, 2009, Hong Kong, China.

vID	date	patentID	diagnosis
1	20080201	3243	Brain Neoplasm
2	20080202	4722	Stomatitis
3	20080202	2973	Tumor of the Uvea
4	20080204	9437	Corneal Intraepithelial Neoplasia
5	20080205	2437	Choroid Tumor
...	...	...	...

Figure 1: The table `visit` recording patient visits

ing semantic queries in database systems. Our experience tells us that if the semantic query can be expressed using traditional query languages such as SQL, processing the semantic query is relatively straightforward even though the challenge of query optimization remains. The crux of the problem lies in the difficulty and complexity of expressing the semantic queries. There are two fundamental challenges in expressing semantic queries. First, ontologies are inherently graph-structured and expressing graph structured queries succinctly is extremely difficult. Second, there is often a mismatch between the semantics in the users mind and the semantics expressed in the ontology. Consequently semantic queries are often difficult to be crisply defined. One possible solution is to define new query languages that would allow the user to express graph-structured queries more easily. Even then, the second challenge is not addressed. The future success of incorporating ontologies into practical database query processing depends on whether we can find automatic or semi-automatic methods to help users express semantic queries.

Our investigation into the problem of expressing semantic queries lead us to find a different approach that would insulate the users from the complexity of the ontology, yet still enable them to ask every possible semantic query. We believe that a semi-automatic framework is required that would bridge the gap between a query in a user's mind and the final result of the query. Furthermore, the users should not be required to handle the ontology directly, or to map the ontology into a relational form.

The rest of the paper is organized as follows. The next two sections describe the two fundamental challenges in detail. Section 4 summarizes the requirements needed to address the two challenges. Section 6 concludes with an outline of a possible approach.

2. CHALLENGE #1: EXPRESSING GRAPH STRUCTURED QUERIES

The first fundamental challenge is the difficulty in expressing queries against a graph structured ontology. The following example illustrates the complexity of expressing semantic queries in SQL.

EXAMPLE 1 (RUNNING EXAMPLE). *Suppose we have a table of patient visit records as shown in Fig. 1, of which the diagnosis column is associated with the NCI Thesaurus ontology (Fig. 2). Consider the query to find all patients diagnosed with eye tumor.*

Using existing RDF-like data models [18], we could store the ontology as triples in the `Thesaurus(src,rel,tgt)` relation and attempt to write the query in Example 1 using recursive SQL:

```
WITH Traversed (src) AS (
   (SELECT src
   FROM Thesaurus
   WHERE tgt = 'Eye Tumor' AND rel='Synonym')
UNION ALL
   §SELECT CH.tgt
   FROM Traversed PR, Thesaurus CH
   WHERE PR.src = CH.src AND CH.rel='is_a'))
SELECT DISTINCT V.*
FROM Thesaurus T, Visit V
WHERE src IN
   (SELECT DISTINCT src FROM Traversed)
   AND T.rel = 'Synonym'
   AND T.tgt = V.diagnosis
```

Alternatively, if we write the same query against the original XML format of the NCI Thesaurus, we have the following.

```
WITH Traversed (cls,syn) AS (
   (SELECT R.cls, R.syn
   FROM XMLTABLE ('Document("Thesaurus.xml")
         /terminology/conceptDef/properties
         [property/name/text()="Synonym" and
         property/value/text()="Eye Tumor"]
         /property[name/text()="Synonym"]/value'
         COLUMNS
         cls CHAR(64) PATH './parent::*/parent::*
                              /parent::*/name',
         tgt CHAR(64) PATH'.') AS R)
UNION ALL
   §SELECT CH.cls,CH.syn
   FROM Traversed PR,
         XMLTABLE ('Document("Thesaurus.xml")
         /terminology/conceptDef/definingConcepts/
         concept[./text()=$parent]/parent::*/parent::*/
         properties/property[name/text()="Synonym"]/value'
         PASSING PR.cls AS "parent"
         COLUMNS
         cls CHAR(64) PATH './parent::*/
                              parent::*/parent::*/name',
         syn CHAR(64) PATH'.') AS CH))
SELECT DISTINCT V.*
FROM Visit V
WHERE V.diagnosis IN
   (SELECT DISTINCT syn FROM Traversed)
```

In both instances, it is not straight-forward to write the query and the user needs to have an intimate knowledge of the structure of the ontology such as existence of "synonym", and "is_a" edges.

3. CHALLENGE #2: FUZZINESS OF SEMANTIC QUERIES

The second fundamental challenge is the inherent fuzziness in the semantics of the query. In most practical applications, the data and the ontology behind it are quite complicated and consequently the queries are no longer exact, that is, users often have no more than a vague notion, rather than a clear understanding and definition, of what they query for. In other words, even if the users have intimate knowledge of the structure of the ontology, they might not be able to precisely specify what they want to find.

EXAMPLE 2. *Find all patients diagnosed with some disease in the choroid, which is part of the eye.*

Intuitively, the user wants to find patients with some disease that affects or is located in the choroid. In the NCI Thesaurus, there are 3 separate types of relationship linking disease concepts to anatomic locations:

1. Disease_Has_Primary_Anatomic_Site
2. Disease_Has_Associated_Anatomic_Site
3. Disease_Has_Metastatic_Anatomic_Site

Even if the user knows the structure of the NCI Thesaurus ontology, i.e., she knows about the three types of relationships that are relevant to the query, without looking at the results, the user still may not know whether the query should use one of these relationships to "choroid" or all of them. If the user does not know the structure of the ontology at all, then he certainly would not know how to specify the query exactly. The query semantics is certainly not crisp and hence not easily expressed in SQL especially when the structure of the ontology is not well-known to the user.

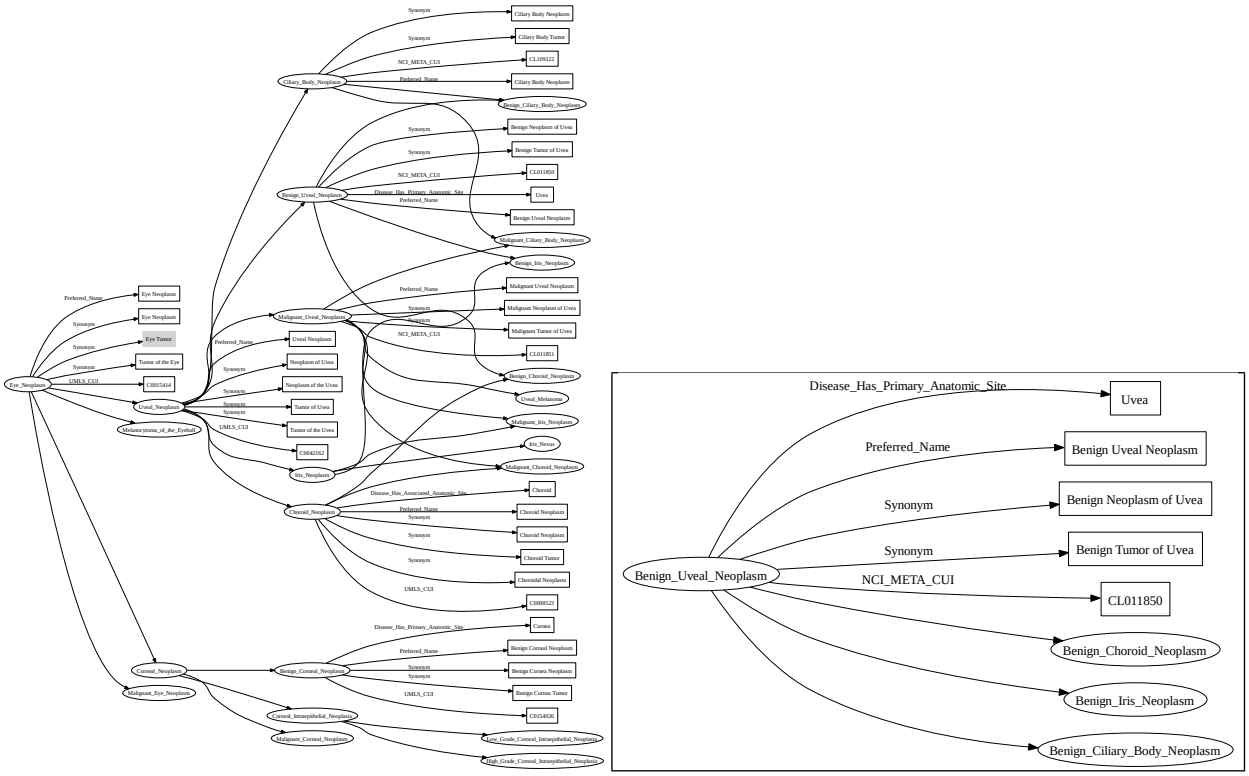

Figure 2: A three level fragment of the NCI Thesaurus ontology. Elliptical nodes denote concepts. Rectangular nodes denote property values. Edges between concepts denote subsumption relationships. Magnified view on the right.

4. ADDRESSING THE CHALLENGES

Our investigation lead us to explore semi-automatic approaches for helping the user to express the semantic queries. In order to address the two challenges we identified, an ideal semi-automatic method needs to satisfy the following requirements.

First, the user should not need to know the structure of the ontology intimately. Ontologies are complex representations of knowledge that are meant for computer processing. It would defeat the purpose of having machine readable ontologies, if the user still needs to understand and know the ontologies.

Second, the user should not need to know how the ontology is modeled and stored in the database. The user should be able to pose the semantic query independent of how the ontology is stored and modeled in the database.

Third, we do not need another query language. Having the user learn another query language and express her semantic queries in it is not reducing the complexity of the task.

Fourth, the user needs to be able to refine the semantic query to match her information needs. As we have previously shown, semantic queries are rarely crisply defined.

One particular semi-automatic approach that we are investigating is using the query by example (QBE) paradigm for semantic queries. The QBE paradigm takes advantage of user-provided examples that satisfy the query as seeds to automate the query process. Specifically, we envision the semantic QBE approach as consisting of three steps. In the first step, the user provides several examples that satisfy the query she has in mind. In the second step, the system learns the semantics of the query from the given examples and their related ontologies. In the third step, the system applies the semantics on the data to generate the entire query result and return it to the user. In the query processing process, as an optional active learning mechanism, the user will be probed systematically to determine whether certain tuples satisfy her query in order to help clarify her intention and speed up the query processing process. False positive errors in the query results can be detected by the user and fed back to the system as part of the active learning process. Since a semantic query is inherently fuzzy, the user typically expects only a subset of the full results, false negative errors can be ameliorated by doing active learning until the desired number of result tuples are obtained.

Using the QBE approach for Example 1, the user no longer needs to write the unwieldy SQL queries, but instead provides examples of tuples, say tuples with vID 4 & 5, that satisfy the query she has in mind. In other words, our method insulates the user entirely from the complexity of understanding and using ontologies. Consequently, there is no need to map ontologies into relational form.

5. RELATED WORK

Managing ontology data alone is not a new topic and several systems have been developed [13, 15, 16, 19, 20] during the past years. Some of these systems store ontology data in a file system, making querying them very hard [16]. The other systems transform the ontology data into RDF form and store the RDF triples in a relational database. Processing of ontology-related queries in these systems is typically done by an external middle-ware (wrapper) layer built on top of a DBMS engine, and DBMS users can't really reference ontology data directly.

Querying relational data together with their semantics encoded in ontology is an emerging topic that attracts a lot of attentions recently. Das et al. [6] proposed a method to support ontology-based

semantic matching in RDBMS using SQL directly. Ontology data are pre-processed and stored in a set of system-defined tables. Several special operators and a new indexing scheme are introduced. A database user can thus reference the ontology data directly using the new operators. The main drawback of their approach is that semantic queries involving the ontology data are usually hard to write and costly to process (in terms of both processing time and storage overhead) due to the graphical structure of the ontology data and the need for reasoning (i.e., transitive closure computation) on the ontology data.

In [12], *virtual view* is proposed as a way to represent relational data together with their related ontology data in a relational view. However, there are three requirements to apply the virtual view idea: (a)language extensions to SQL to support the creation and use of the virtual view, (b) the DBMS engine must support native XML data (together with relational data) and the processing of the virtual view related operators, and (c) to create a virtual view, the user must understand the complex ontology data and their relationship with the base relational data completely.

Query by Example (QBE) is a well-known concept in database community. It was first proposed by Moshé M. Zloof in the mid 1970s [21, 22] as a query language that can be used by database users to define and query a relational database. It is quite different from SQL in that it is a graphical query language. Its interface is usually virtual tables where the user could enter commands, examples, etc. After QBE was presented, most research work around QBE has been focused on enrichment and extension of QBE as a query language and developing efficient methods for generating and processing the queries defined by the examples [10].

In commercial database products, QBE is widely used as graphical front-end for RDBMSs [1]. It is also used as a convenient interface for users to specify queries for image, video, and document databases, and various techniques have been studied [2, 3, 4, 17].

There are two common characteristics for all the previous QBE work: (a) the examples are used to specify a query that will be generated (b) the generated query is a "normal" query in terms that all the query conditions (may be in the forms of similarity measures) are defined on the *base attributes* in the underlying tables.

The semantic QBE problem we discussed in their paper is very different from the traditional QBE problem. First, due to the complexity of the semantic information associated with the data in the base relational tables, the real query associated with the user's intention which is specified by the input examples is really hard (or impossible) to capture by a traditional SQL query. Secondly, in the problem we described, the underlying "query" is defined not only on the base attributes in the relational table, but also on the semantics of the base data encoded in the ontology and the connections between the relational data and the ontology data.

6. CONCLUSION

In this paper, we have argued for the importance of the problem of expressing semantic queries and identified two fundamental challenges, namely, the complexity of expressing graph structured queries, and the fuzziness of semantic queries. To address these two challenges, we believe that semi-automatic approaches are need to help user formulate semantic queries and we outlined several important requirements. We also propose to use the query by example (QBE) paradigm for semantic queries, because it addresses the requirements we identified. As future work we plan to investigate the concrete algorithms to realize the QBE paradigm for semantic queries.

7. REFERENCES

[1] A. Balter. *Mastering Microsoft Office Access 2007 Development*. Sams, 2007.

[2] M. Belkhatir, P. Mulhem, and Y. Chiaramella. A conceptual image retrieval architecture combining keyword-based querying with transparent and penetrable query-by-example. In *ACM International Conference on Image and Video Retrieval (CIVR)*, pages 528–539, 2005.

[3] G. Boccignone, A. Chianese, V. Moscato, and A. Picariello. Animate system for query by example in image databases. In *EuroIMSA*, pages 451–456, 2005.

[4] A. K. Choupo, L. Berti-Equille, and A. Morin. Optimizing progressive query-by-example over pre-clustered large image databases. In V. Benzaken, editor, *BDA*, 2005.

[5] ASTM E2369-05 Standard Specification for Continuity of Care Record (CCR). `http://www.astm.org`.

[6] S. Das, E. I. Chong, G. Eadon, and J. Srinivasan. Supporting ontology-based semantic matching in RDBMS. In *Proc. of Very Large Database (VLDB)*, pages 1054–1065, 2004.

[7] Health level seven. `http://www.hl7.org`.

[8] International Classification of Diseases, Ninth Revision, Clinical Modification (ICD-9-CM). `http://www.cdc.gov/nchs/about/otheract/icd9/abticd9.htm`.

[9] Systematized Nomenclature of Medicine-Clinical Terms. `http://www.ihtsdo.org/`.

[10] R. Krishnamurthy, S. P. Morgan, and M. M. Zloof. Query-by-example: Operations on piecewise continuous data (extended abstract). In *Proc. of Very Large Database (VLDB)*, pages 305–308, 1983.

[11] L. Lim, H. Wang, and M. Wang. Semantic data management: Towards querying data with their meaning. In *Int. Conf. Data Engineering (ICDE)*, pages 1438–1442, 2007.

[12] L. Lim, H. Wang, and M. Wang. Unifying data and domain knowledge using virtual views. In *Proc. of Very Large Database (VLDB)*, pages 255–266, 2007.

[13] L. Ma, Z. Su, Y. Pan, L. Zhang, and T. Liu. RStar: An RDF storage and query system for enterprise resource management. In *Intl' Conf. on Information and Knowledge Management (CIKM)*, 2004.

[14] National cancer institute thesaurus. `http://www.nci.nih.gov/cancerinfo/terminologyresources`.

[15] OntoBroker. `http://ontobroker.aifb.uni-karlsruhe.de/index_ob.html`.

[16] OTK tool repository: Ontoedit. `http://www.ontoknowledge.org/tools/ontoedit.shtml`.

[17] N. Rasiwasia, N. Vasconcelos, and P. J. Moreno. Query by semantic example. In *ACM International Conference on Image and Video Retrieval (CIVR)*, pages 51–60, 2006.

[18] SWAD-Europe Deliverable 10.2: Mapping Semantic Web Data with RDBMSes. `http://www.w3.org/2001/sw/Europe/reports/scalable_rdbms_mapping_report/`.

[19] The KArlsruhe ONtology and semantic web tool suite. `http://kaon.semanticweb.org/`.

[20] The protégé ontology editor and knowledge acquisition system. `http://protege.stanford.edu/`.

[21] M. M. Zloof. Query-by-example: the invocation and definition of tables and forms. In D. S. Kerr, editor, *Proc. of Very Large Database (VLDB)*, pages 1–24. ACM, 1975.

[22] M. M. Zloof. Query-by-example: A data base language. *IBM Systems Journal*, 16(4):324–343, 1977.