Optimizing Data Integration Queries Over Web Data Sources (OPTIQ)

Muhammad Intizar Ali
Database and Information Systems Group,
Department of Computer Science, COMSATS Institute of Information Technology,Lahore.
intizarali@ciitlahore.edu.pk

Keywords: Data Integration: Distributed Query Processing: Query Optimization: XSPARQL: SPARQL: XQUERY: RDF: XML: Web Data Sources.

Abstract: Nowadays data integration must deal with a far more in-congruent environment than in the pre-Web era. There are multiple efforts to design new query languages (which we hereafter call "data integration query languages"), that combine the features of two or more existing well established query languages to integrate data. However modern data integration applications still face many challenges while executing data integration queries over web data sources. Currently data integration queries are just focused on data integration while optimized query plan generation for such queries has not been extensively studied. Higher query processing time, data shipping and transformation on the fly, complicated syntax, efforts required to learn a new query language and limitation imposed on query languages are also a big hindrance in the emergence and wide adoption of the data integration query languages.

In this paper, we have optimized data integration queries by parallel and distributed query processing over heterogeneous web data sources. We combine the data integration capabilities of the XSPARQL (a data integration query language for XML and RDF data sources) with the distributed and parallel query execution capabilities of DeXIN (a framework for highly distributed large web data sources integration).

Parallel query processing suits very well for data integration of web data sources because of the highly distributed nature of the web. Experimental results are also evident of better performance of optimized query processing by parallelizing the data integration query processing. In the later part of this paper we argue that availability of an easy to use visual editor for data integration queries will greatly help in wide adoption and usage of data integration query language for web data sources integration.

1 INTRODUCTION

In modern business enterprises, it is frequent to develop an integrated application to provide uniform access to multiple existing information systems running internally or externally of the enterprise. Data integration is a pervasive challenge faced in these applications that need to query across multiple autonomous and heterogeneous data sources. Integrating such diverse information systems becomes a challenging task particularly when different applications use different data formats and query languages which are not compatible with each other. With the growing popularity of web technologies and availability of the huge amount of data on the web, the requirements for data integration has changed from the traditional database integration approaches. The large scale of Web has led to high levels of distribution, heterogeneity, different data formats and query languages.

Data transformation and query rewriting are common approaches to mitigate the heterogeneity among various data formats [8]. Ample efforts are made to transform the data sources from one data format to another data format [9, 10]. However data transformation approaches are not viable for large or dynamic data sources, because the frequency of changes might make it cumbersome to perform the transformation over and over again. Another

approach is to rewrite query from one query language to another query language while data sources maintain their original data format [1]. In the past data, transformation from relational data to XML and query rewriting from SQL to XQUERY attracted significant research while nowadays data transformation from XML to RDF and query rewriting from XQUERY to SPARQL and vice versa is likewise being explored [7]. XSPARQL and embedding SPARQL into XSLT are data integration query languages to integrate XML and RDF data [1, 10]. The DeXIN framework is another query-side effort to integrate distributed heterogeneous Web Data sources [2]. However complex syntax, inability to generate optimized solutions and efforts required to learn a new language were prominent obstacles in the adoption of these projects from academia to industry. Hence, unable to leverage the modern multi-core processing capabilities, which are of utmost important for efficient query processing over highly distributed Web data sources.

In this paper we optimize data integration queries of XSPARQL by embedding XSPARQL queries into DeXIN framework. We categorize various features of data integration queries and then combine them to get the best query execution plan for distributed query processing over heterogeneous distributed web data sources. We perform experimental evaluation and compare query execution time of the optimize query plan with the existing data integration query languages processing time. It is evident from the experimental results that our approach performs very well while executing distributed query over multitude of data sources scattered over Web.

Our main focus is to devise strategies and algorithms for optimized query plan generation for data integration queries specifically targeted at Web data integration scenarios where data integration applications lack prior knowledge or control of the content of data sources. Optimized query plans for data integration queries over the Web of Data should be capable of dealing with exponential growth and continuous updates in terms of information sources and dynamic streaming data. Query optimization should also take into account scalability issues to avoid performance degradation and inefficient query execution plans.

In order to attract the wider community of users we focus on enabling optimized data integration query generation using visual editors and initiation of an open source data integration suite for distributed, parallelized and heterogeneous querying over Web data sources.

Rest of the paper is organized as: in Section 2, we briefly describe data integration queries for SPARQL and XQuery integration. We compare the features of various data integration queries to categorize and enhance existing data integration queries. In Section 3, we propose optimize query execution plan for data integration queries by embedding XSPARQL into DeXIN framework. Later in this section we discuss possible enhancement in the features of existing data integration queries. Experimental evaluation is performed in Section 4. We conclude this paper with possible future work on data integration queries in Section 5.

2 DATA INTEGRATION QUERY LANGUAGES

Data integration query languages combine the features of two or more existing query languages to integrate data on the fly. In this approach a new query language is designed or already existing query language is extended in order to query multiple heterogeneous web data sources in their respective query languages. Data integration query language is a broader term, which can combine any arbitrary two or more query languages features. However in this paper we restrict our self to those data integration query languages, which combine the features of XQuery and SPARQL. Data integration query language approach is a viable solution for Web data sources integration, which not only integrates data on the fly but also can easily manage rapidly changing web data sources. In this section we will briefly describe three prominent approaches of data integration queries for XML and RDF data sources and then compare their features to highlight the advantages and disadvantages of the each approach.

2.1 Embedding SPARQL into XQUERY

Embedding SPARQL into XQUERY is one of the initiative efforts for designing data integration query to provide a uniform access over RDF and XML data sources [10]. The approach is to select one query language for one data format as a base query language and embed the other query language for another format into the base query language. In embedding SPARQL into XQUERY approach, all SPARQL queries are embedded into XQuery/XSLT

and automatically transformed into pure XQuery/XSLT queries to be posed against pure XML data after transformation of RDF data into XML. This embedding enables users to benefit from graph and tree language constructs of both SPARQL and XQuery. The authors defined a formal SPARQL algebra to transform a SPARQL query into an operator tree of SPARQL algebra. The operator tree is later translated into XQuery/XSLT. Table 1 shows an example XQuery with SPARQL embedded inside XQuery.

The work presented in [10] is not an exclusive work on SPARQL embedded into XQuery. There exists some other similar work with slightly different approach also available in the literature. Contrary to embedding SPARQL into XQuery/XSLT, there are also some efforts to embed XPATH and XQuery into SPARQL queries [9].

Table 1: An example XQuery with SPARQL embedded inside XQuery.

```
( 1 ) d e c l a r e namespace f o a f =" h t t p : / / xmlns .
com/ f o a f / 0 . 1 / " ;
( 2 ) < r e s u l t s >{ f o r ( $n , $m) i n
( 3 ) SELECT ?name ?mbox
( 4 ) WHERE { ? x f o a f : name ?name .
( 5 ) ? x f o a f :mbox ?mbox .
( 6 ) FILTER r e g e x ( s t r ( ?mbox ) , "@work . example " ) }
( 7 ) r e t u r n <result><name>{$n}</name><mbox>{$m}
( 8 ) </mbox> </ r e s u l t >}< r e s u l t s >
```

2.2 XSPARQL

XSPARQL is a new query language or more precisely to say data integration query language designed with the idea to combine XQUERY and SPARQL query languages. Despite the availability of GRDDL transformation sets the translating between XML and RDF is a tedious and error prone task [8]. In [1], a new query language based approach is used to transform XML into RDF and vice versa. XSPARQL is a query language combining XQuery and SPARQL for transformations between RDF and XML. XSPARQL subsumes XQuery and most of SPARQL (excluding ASK and DESCRIBE). Conceptually, XSPARQL is a simple merge of SPARQL components into XQuery. XQuery is a native query language in XSPARQL and all XQuery queries are also considered as XSPARQL queries. In order to execute SPARQL queries inside the body of XQuery, the XQuery FLWOR expression is slightly modified which is called FLWOR' expression. Concerning semantics, XSPARQL equally builds on top of its constituent languages. They extended the formal semantics of XQuery by additional rules which reduce each XSPARQL query to XQuery expressions; the resulting FLWORs operate on the answers of SPARQL queries in the SPARQL XML result format. All XSPARQL queries are rewritten in XQuery standard format while SPARQL queries are executed over SPARQL endpoints and results are returned in RDF/XML.

2.3 Distributed Extended XQuery for Data Integration (DeXIN)

DeXIN is an extensible framework for providing integrated access over heterogeneous, autonomous, and distributed web data sources, which can be utilized for data integration in modern web applications and service oriented architecture. DeXIN extends the XQuery language by supporting SPARQL queries inside XQuery, thus facilitating the query of data modelled in XML, RDF, and OWL [3]. DeXIN facilitates data integration in a distributed web and service oriented environment by avoiding the transfer of large amounts of data to a central server for centralized data integration and avoids the transformation of a huge amount of data into a common format for integrated access.

At the heart of DeXIN is an XQuery extension that allows users/applications to execute a single query against distributed, heterogeneous web data sources or data services. DeXIN considers one data format as the basis (the so-called "aggregation model") and extends the corresponding query language to executing queries over heterogeneous data sources in their respective query languages.

Currently, DeXIN have implemented XML as an aggregation model and XQuery as the corresponding language, into which the full SPARQL language is integrated. However, this framework is very flexible and could be easily extended to further data formats (e.g., relational data to be queried with SQL) or changed to another aggregation model (e.g., RDF/OWL rather than XML).

The main highlights of the features of the DeXIN are as follows.

- DeXIN is an extensible framework for parallel query execution over distributed, heterogeneous and autonomous large data sources.
- DeXIN provides extension of XQuery which covers the full SPARQL language and supports the decentralized execution both XQuery and SPARQL in a single query.

- DeXIN approach supports the data integration of XML, RDF and OWL data without the need of transforming large data sources into a common format.
- DeXIN is implemented as a web service to provide easy access using service oriented architecture.
- DeXIN can easily be integrated into existing web applications as a data integration tool.
- Experimental results show good performance and reduced network traffic achieved with DeXIN approach.

2.3 Comparing Features of Data Integration Queries

In Table 2, we compare different features of the various data integration queries designed to integrate XML and RDF data sources using combination of XQuery and SPARQL query languages. All the three approaches are mainly intended for data integration of XML and RDF. Embedded SPARQL approach performs poorly when the transformation of the large data sources is required while XSPARQL and DeXIN only transform results into uniform format rather than whole data source. Both Embedded SPARQL and XSPARQL require its user to learn a new query language while DeXIN does not make merely noticeable syntactic changes to the participating query languages. DeXIN also executes parallel and distributed queries for distributed web data sources. Variable induction is a strong feature of XSPARQL which enables it to share same variable for joining two or more heterogeneous Web data sources within a single query.

3 OPTIMIZING DATA INTEGRATION QUERIES

Existing data integration query languages are mainly focused on data integration while optimize query execution plan for such query language have not been studied. In this section we propose the combination of various features of the existing data integration query language which not only be utilized for optimize query execution plan but also can better cope with the modern data integration challenges faced because of the highly distributed and heterogeneous nature of the Web, availability of large amount of data and rapidly changing data sources over Web.

Table 2: Comparison of various features of data integration queries for XML and RDF data

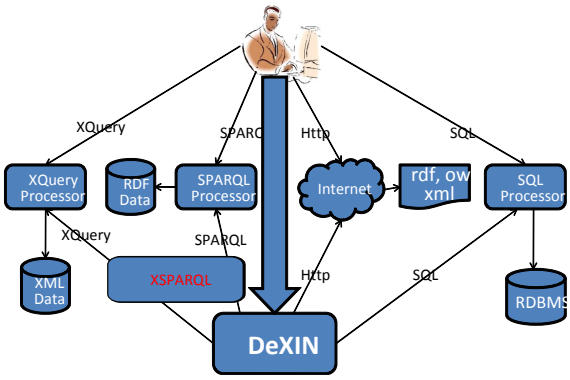
Features	**Embedded SPARQL**	**XSPARQL**	**DeXIN**
Data integration	Yes	Yes	Yes
Data transformation	Yes	No	No
Complicated syntax	Yes	Yes	No
Parallel/distributed query execution	No	No	Yes
Visual interface	No	No	Yes
Restriction imposed on query language	Yes	Yes	No
Variable induction	No	Yes	No
Query language definition	No	Yes	No
Flexibility for enhancement to further query languages	No	No	Yes

3.1 Parallel and Distributed Query Processing

As a first step towards the solution of the optimized query processing of data integration queries, we integrate the DeXIN framework with XSPARQL. Figure 1 shows the integration of XSPARQL into DeXIN framework. This will combine the data integration capabilities of XSPARQL with the distributed and parallel query execution capabilities of DeXIN. Optimized query plan generation for the Web of Data is integral to Web data sources integration. We devise strategies and algorithms for optimized query plan generation for data integration queries specifically targeted at Web data integration scenarios where data integration applications lack prior knowledge or control of the content of data sources. Optimized query plans for data integration queries over the Web of Data should be capable of dealing with exponential growth and continuous updates in terms of information sources and dynamic streaming data. Query optimization should also take into account scalability issues to avoid performance degradation and inefficient query execution plans.

3.2 SPARQL and XQuery Endpoints

Figure 1: Parallel XSPARQL Query Processing with DeXIN



XSPARQL and embedded SPARQL deal with the RDF and XML data source as a flat file. The general procedure followed in these applications is to fetch the RDF/XML file, create an in-memory tree/graph and then execute queries. This procedure will become bottle neck once the size of the data source increases. Nowadays many XML and RDF databases are available, which are specifically designed to efficiently store and process Web data sources. On top of these databases SPARQL/ XQuery endpoints are available which provide direct access to the user for executing queries over these data sources. After the integration of XSPARQL queries inside DeXIN framework, data integration queries can benefit from endpoints of SPARQL and XQuery.

3.3 Dynamic Data Source Selection

Data source selection is an important task while integrating distributed heterogeneous web data sources because data sources are discovered, selected and processed dynamically without any prior knowledge of the data sources. DeXIN keeps tracks of data services and their statistics for best data source selection. Moreover, DeXIN also stores user's concerns using profiling approach which helps to select the most appropriate data source for that particular user if multiple data sources are available to perform the same task [4, 5].

3.4 Visual Editor

Complicated syntax, efforts required to learn a new query language and limitation imposed on query languages are also a big hindrance in the emergence and wide adoption of the data integration query languages. We propose a query-based aggregation of multiple heterogeneous data sources by combining powerful querying features of XQuery and SPARQL with an easy interface of a mashup tool for data sources in XML and RDF. Our mashup editor allows for automatic generation of mashups with an easy to use visual interface. We utilize the concept of data mashups and use it to dynamically integrate heterogeneous web data sources by using the extension of XQuery proposed in the DeXIN. All available data sources over the Internet are considered as a huge database and each data source is considered as a table. Data mashups can generate queries in extended XQuery syntax and can execute the sub-queries on any available data source contributing to the mashup [6]. We aim to design a visual editor for XSPARQL as well which can automatically generate queries from visual interface.

4 EXPERIMENTAL EVALUATION

Testbed: We have implemented parallel and distributed query processing for XSPARQL queries. For prototype development we used java, Saxon is used for XQuery processing and ARQ is used for SPARQL query processing. Our testbed includes 3 computers (Intel(R) Core(TM) 2 CPU, 2.4 GHz, 4GB RAM) running SUSE Linux with kernel version 2.6. The machines are connected over a standard 100Mbit/S network connection. An open source native XML database eXist (release 1.2.4) is installed on each system to store XML data.

Data Sets: For the evaluation of our implementation we used the XMark benchmark suite, which is the most widely used benchmark suite for XQuery [11]. XMARK benchmark suite is bundled with a data generator that produces XML data sets. The benchmark also includes a set of 20 XQuery queries over this generated data. Using the provided data generator, we created XML datasets with sizes of 1, 2, 10, 50 and 100MB. For RDF data sets we rely on XMARK and converted these XML data sets of XMARK into RDF using XSPARQL. We reformulated XMARK 20 queries into SPARQL,

XSPARQL, Embedded SPARQL and DeXIN extended XQuery format.

Experimental Evaluation:

Figure 2 shows experimental result of evaluating first 10 queries of XMARK benchmark executed over data set of 2 MB. Embedded SPARQL performs better when the data size is small while XSPARQL performs well with bigger data size. However XSPARQL performs very poorly when queries contain nested join between two data sets. DeXIN improves the performance of XSPARQL particularly when there are multiple data sets are involved in query processing.

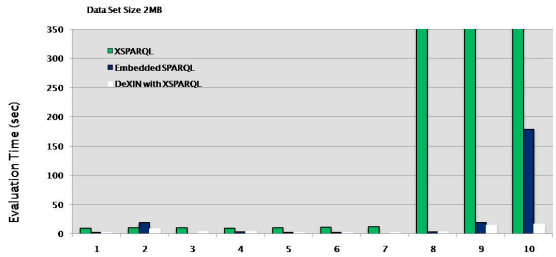


Figure 2: Query execution time comparison

5 CONCLUSION AND FUTURE WORK

In this paper we have integrated XSPARQL and DeXIN capabilities to execute data integration queries in parallel. Experimental evaluation has shown promising results. By integrating XSPARQL inside DeXIN the performance of data integration queries (particularly those queries which involve highly distributed multiple data sources) has greatly increased. We believe that our proposed plan will lead to the optimized query execution plans for the data integration queries over the Web. Our framework will act as a basis for the development of data integration applications over distributed heterogeneous and continuously updating Web data sources. A visual tool to generate data integrating queries will attract a wider spectrum of the users to create data integration applications on the fly.

REFERENCES

[1] Waseem Akhtar, Jacek Kopecký, Thomas Krennwallner, and Axel Polleres. XSPARQL: Traveling between the XML and RDF Worlds - and Avoiding the XSLT Pilgrimage. In Proc. ESWC 2008, volume 5021 of Lecture Notes in Computer Science, pages 432–447.Springer, 2008.

[2] Muhammad Intizar Ali, Reinhard Pichler, Hong Linh Truong, and Schahram Dustdar. DeXIN: An Extensible Framework for Distributed XQuery over Heterogeneous Data Sources. In Proc. ICEIS 2009, volume 24 of Lecture Notes in Business Information Processing, pages 172–183. Springer, 2009.

[3] Muhammad Intizar Ali, Reinhard Pichler, Hong Linh Truong, and Schahram Dustdar. On Using Distributed Extended XQuery for Web Data Sources as Services. In Proc. ICWE 2009, volume 5648 of Lecture Notes in Computer Science, pages 497–500. Springer, 2009.

[4] Muhammad Intizar Ali, Reinhard Pichler, Hong Linh Truong, and Schahram Dustdar. Data Concern Aware Querying Using Data Services. In Proc. ICEIS 2011, pages 111–119. SciTePress, 2011.

[5] Muhammad Intizar Ali, Reinhard Pichler, Hong Linh Truong, and Schahram Dustdar. Incorporating Data Concerns into Query Languages for Data Services. to appear in Lecture Notes in Business Information Processing. Springer, 2011.

[6] Muhammad Intizar Ali, Reinhard Pichler, Hong Linh Truong, and Schahram Dustdar. On Integrating Data Services Using Data Mashups. In Proc. BNCOD 2011, volume 7051 of Lecture Notes in Computer Science. Springer, 2011.

[7] Nikos Bikakis, Nektarios Gioldasis, Chrisa Tsinaraki, and Stavros Christodoulakis. Querying XML Data with SPARQL. In Proc. DEXA 2009, volume 5690 of Lecture Notes in Computer Science, pages 372–381. Springer, 2009.

[8] Dan Connolly. Gleaning Resource Descriptions from Dialects of Languages (GRDDL). W3C Recommendation, W3C, September 2007. http://www.w3.org/TR/2007/REC-grddl-20070911/.

[9] Matthias Droop, Markus Flarer, Jinghua Groppe, Sven Groppe, Volker Linnemann, Jakob Pinggera, Florian Santner, Michael Schier, Felix Schöpf, Hannes Staffler, and Stefan Zugal. Embedding XPath Queries into SPARQL Queries. In Proc. ICEIS 2008, pages 5–14, 2008.

[10] Sven Groppe, Jinghua Groppe, Volker Linnemann, Dirk Kukulenz, Nils Hoeller, and Christoph Reinke. Embedding SPARQL into XQuery/XSLT. In Proc. SAC 2008, pages 2271–2278, 2008.

[11] Schmidt, A., Waas, F., Kersten, M.L., Carey, M.J., Manolescu, I., Busse, R.: XMark: A Benchmark for XML Data Management. In: Proceedings of the 28th international conference on Very Large Data Bases. pp. 974–985 (2002).