Web Semantics: Science, Services and Agents on the World Wide Web 37–38 (2016) 90–111

Contents lists available at ScienceDirect

Web Semantics: Science, Services and Agents on the World Wide Web

journal homepage: www.elsevier.com/locate/websem

RIQ: Fast processing of SPARQL queries on RDF quadruples

Anas Katib, Vasil Slavov, Praveen Rao *

Department of Computer Science and Electrical Engineering, University of Missouri-Kansas City, Kansas City, MO, USA

ARTICLE INFO

Article history:
Received 9 April 2015
Received in revised form
31 January 2016
Accepted 16 March 2016
Available online 5 April 2016

Keywords:
RDF
Quadruples
SPARQL
Query processing
Knowledge graphs

ABSTRACT

In this paper, we address the problem of fast processing of SPARQL queries on a large RDF dataset, where the RDF statements are quadruples (or quads). Quads can capture provenance or other relevant information about facts. This is especially powerful in modeling knowledge graphs, which are becoming increasingly important on the Web to provide high quality search results to users. We propose a new approach called RIQ that employs a *decrease-and-conquer* strategy for fast SPARQL query processing. Rather than indexing the entire RDF dataset, RIQ identifies groups of similar RDF graphs and creates indexes on each group separately. It employs a new vector representation for RDF graphs and locality sensitive hashing to construct the groups efficiently. It constructs a novel filtering index on the groups and compactly represents the index as a combination of Bloom and Counting Bloom Filters. During query processing, RIQ employs a streamlined approach. It constructs a query plan for a SPARQL query (containing one or more graph patterns), searches the filtering index to quickly identify candidate groups that may contain matches for the query, and rewrites the original query to produce an optimized query for each candidate. The optimized queries are then executed using an existing SPARQL processor that supports quads to produce the final results. We conducted a comprehensive evaluation of RIQ using a real and synthetic dataset, each containing about 1.4 billion quads. Our results show that RIQ can outperform its competitors designed to support named graph queries on RDF quads (e.g., Jena TDB and Virtuoso) for a variety of queries.

1. Introduction

The Resource Description Framework (RDF) is a standard model for data representation and interchange on the Web [1]. Today, RDF uses IRIs to name entities and their relationships. It enables easy merging of different data sources. While RDF was introduced in the late 90s as the data model for the Semantic Web, only in recent years, it has gained popularity on the Web. For example, Linked Data [2] exemplifies the use of RDF on the Web to represent different knowledge bases (*e.g.*, DBpedia [3]). Another example is Wikidata [4], a sister project of Wikipedia, which publishes facts in RDF. Advanced RDF technologies provide the ability to conduct semantic reasoning in domains such as biopharmaceuticals, defense and intelligence, and healthcare. Several companies have adopted Semantic Web technologies for different use cases such as data aggregation (*e.g.*, Pfizer [5]), publishing datasets on the Web and providing better quality search results (*e.g.*, Newsweek, BBC, The New York Times, Best Buy) [6].

Another important use case of RDF is in the representation of knowledge graphs, which are emerging as a key resource for companies like Google [7], Facebook [8], and Microsoft [9] to provide higher quality search results and recommendations to users. Essentially, a knowledge graph is a collection of entities, their properties, and relationships among entities. Using SPARQL [10], queries can be posed on these knowledge graphs.

In RDF, a fact or assertion is represented as a (subject, predicate, object) triple. A set of RDF triples can be modeled as a directed, labeled graph. A triple's subject and object denote the source and sink vertices, respectively, and the predicate is the label of the edge from the source to the sink. An RDF quad is denoted by a (subject, predicate, object, context). The context (a.k.a. graph name) is used to capture the provenance or other relevant information of a triple. This is especially powerful in modeling the facts in a knowledge graph. Moreover, there are datasets and knowledge bases on the Web such as Billion Triples Challenges [11], Linking Open Government Data (LOGD) [12], and Yago [13] which contain over a billion quads. One can view these datasets as a collection of RDF named graphs. Using SPARQL's GRAPH keyword [10], a query can be posed on RDF named graphs to match a specific graph pattern within any single RDF graph.

* Corresponding author.
E-mail addresses: anaskatib@mail.umkc.edu (A. Katib), vgslavov@mail.umkc.edu (V. Slavov), raopr@umkc.edu (P. Rao).

http://dx.doi.org/10.1016/j.websem.2016.03.005
1570-8268/

```
@PREFIX res: <http://dbpedia.org/resource> .
@PREFIX onto: <http://dbpedia.org/ontology> .

res:Oswego onto:areaLand "5438975.0317056"^^<http://www.w3.org/2001/XMLSchema#double> <http://dbpedia.org/data/Oswego.xml> .
res:Oswego onto:areaCode "620"@en <http://dbpedia.org/data/Oswego.xml> .
res:Oswego onto:isPartOf res:Labette_County,_Kansas <http://dbpedia.org/data/Oswego.xml> .
res:Oswego onto:country res:United_States <http://dbpedia.org/data/Oswego.xml> .
res:Oswego onto:postalCode "67356"@en <http://dbpedia.org/data/Oswego.xml> .
res:Oswego onto:utcOffset "-6"@en <http://dbpedia.org/data/Oswego.xml> .
res:Oswego onto:utcOffset "-5"@en <http://dbpedia.org/data/Oswego.xml> .
res:Oswego onto:areaWater "0"^^<http://www.w3.org/2001/XMLSchema#double> <http://dbpedia.org/data/Oswego.xml> .

res:Salamiou onto:abstract "Salamiu - wie\u015B na Cyprze, w dystrykcie Pafos."@pl <http://dbpedia.org/data/Salamiou.xml> .
res:Salamiou onto:timeZone res:Eastern_European_Summer_Time <http://dbpedia.org/data/Salamiou.xml> .
res:Salamiou onto:isPartOf res:Paphos_District <http://dbpedia.org/data/Salamiou.xml> .
res:Salamiou onto:country res:Cyprus <http://dbpedia.org/data/Salamiou.xml> .
res:Salamiou onto:postalCode "6211"@en <http://dbpedia.org/data/Salamiou.xml> .
res:Salamiou onto:utcOffset "+3"@en <http://dbpedia.org/data/Salamiou.xml> .
res:Salamiou onto:utcOffset "+2"@en <http://dbpedia.org/data/Salamiou.xml> .
res:Salamiou onto:populationTotal "255"^^<http://www.w3.org/2001/XMLSchema#integer> <http://dbpedia.org/data/Salamiou.xml> .
```

G_1

G_2

Fig. 1. Dataset D containing RDF quads.

The popularity of the RDF data model coupled with the availability of very large RDF datasets continues to pose interesting technical challenges for storing, indexing, and query processing of RDF data. In this paper, we address the problem of fast processing of SPARQL queries on RDF quads. In recent years, there has been a flurry of interest within the database community to develop scalable techniques for indexing and query processing of large RDF datasets. Several techniques have been proposed for RDF datasets containing triples [14–21], where each triple consists of a subject, predicate, and an object. One may wonder if we can simply ignore the context in a quad and use any of the previous approaches for processing a query with the `GRAPH` keyword. Unfortunately, this may produce incorrect results, because subpatterns of a graph pattern may match RDF terms in different graphs. Furthermore, none of these approaches has investigated how large, complex graph patterns (*e.g.*, containing undirected cycles) in SPARQL queries can be processed efficiently. Evidently, RDF-3X [16], a popular scalable approach for a local environment, yields poor performance when SPARQL queries containing large, complex graph patterns are processed over large RDF datasets [22]. This is because of the large number of join operations that must be performed to process a query. We argue that, on RDF datasets containing billions of quads, any approach that first finds matches for subpatterns in a large graph pattern and then employs join operations to merge partial matches will face a similar limitation.

Motivated by the above reasons, we developed a new tool called RIQ (**R**DF **I**ndexing on **Q**uads) for fast processing of SPARQL queries on RDF quads. The salient features of RIQ are summarized below:

- RIQ adopts a new vector representation for RDF graphs and graph patterns in SPARQL queries. This representation captures the properties of the triples in an RDF graph and triple patterns in a query. It facilitates grouping similar RDF graphs using locality sensitive hashing [23] and building a novel filtering index for efficient query processing. RIQ uses a combination of Bloom Filters and Counting Bloom Filters to compactly store the filtering index. In addition to the filtering index, each group of similar RDF graphs is indexed separately rather than constructing a single index on the entire collection of RDF graphs.
- RIQ employs a streamlined approach to efficiently process a SPARQL query via the *decrease-and-conquer* strategy. Using the filtering index, RIQ quickly identifies candidate groups of RDF graphs that may contain a match for the query. It methodically rewrites the original query and executes optimized queries on the candidates using a conventional SPARQL processor that supports quads (*e.g.*, Jena TDB [24]).
- RIQ achieved high performance on a real-world and synthetic dataset, each containing about 1.4 billion quads, on a variety of SPARQL queries. It yielded superior performance for high selectivity queries that matched a small fraction of the named graphs in a dataset, when I/O was the dominating factor.

A preliminary version of this work appeared in the 17th International Workshop on the Web and Databases (WebDB) 2014 [22].

The rest of the paper is organized as follows. Section 2 provides the background on RDF and SPARQL. Section 3 describes the related work and the motivation of our work. Section 4 describes the novel design of RIQ including the new vector representation of RDF graphs and graph patterns, filtering index construction, and the query processing approach. Section 5 presents the performance evaluation results and comparison of RIQ with its competitors. Finally, we provide our concluding remarks in Section 6.

2. Background and preliminaries

In this section, we provide a brief background on RDF and SPARQL. After that, we describe popular techniques based on hashing that underpin the design of RIQ.

2.1. RDF and SPARQL

The RDF data model provides a simple way to represent any assertion as a (subject, predicate, object) triple. A collection of triples can be modeled as a directed, labeled graph. A triple can be extended with a graph name (or context) to form a quad. Quads with the same context belong to the same RDF graph.

Using SPARQL, one can express complex graph pattern queries on RDF graphs. A triple pattern contains variables (prefixed by `?`) and constants. A Basic Graph Pattern (BGP) in a query combines a set of triple patterns. During query processing, the variables in a BGP are bound to RDF terms in the data, *i.e.*, the nodes in the same RDF graph, via subgraph matching [10]. Common variables within a BGP or across BGPs denote a join operation on the variable bindings of triple patterns. `UNION` combines bindings of multiple graph patterns; `OPTIONAL` allows certain patterns to have empty bindings; `FILTER EXISTS/NOT EXISTS` tests for existence/non-existence of certain graph patterns. The variable `?g` will be bound to the contexts of those RDF graphs that contain a match for the entire set of graph patterns and predicates, if any, inside the `GRAPH` block.

Example 1. Consider the dataset D shown in Fig. 1, which contains two RDF graphs G_1 and G_2. Consider a query Q shown in Fig. 2. It has five BGPs. Consider the pattern BGP_1 in Q. The bindings for the variable `?city` in the triple pattern `?city onto:areaLand ?area` are joined with those for `?city` in `?city onto:areaCode ?code`. If Q is executed on D, `?g` will be bound only to the context of G_1, *i.e.*, `<http://dbpedia.org/data/Oswego.xml>`. Note that BGP_3 does not have a match in G_2 as the only country mentioned in G_2 is Cyprus.