A RULE-BASED SPARQL ENDPOINT WRAPPER

Mamdouh Farouk and Mitsuru Ishizuka
Creative Informatics Department, The University of Tokyo, Tokyo 1138656, Japan

Keywords: RDF, SPARQL Endpoint, SPARQL Query.

Abstract: The web of data is a web vision in which data are represented into RDF and interlinked to each other. The fast growth of linked data motivates researchers to exploit this huge amount of well-represented data in linked data cloud. Moreover, finding the implicit answers is important to enable web agent to understand deeply web data. This paper proposes a SPARQL endpoint wrapper to enable original endpoint to answer more queries and facilitate query writing for the client. The proposed wrapper uses inference rules to expand user query. A prototype for the proposed wrapper is presented to show the effectiveness of the proposed approach.

1 INTRODUCTION

Linked data is a way for publishing web data into a machine understandable format. Resource Description Framework (RDF) is used as standard language for linked data. RDF is a W3C recommendation that represents current web data into machine understandable format. In this way, data will be more useful. Moreover, since the start of linked data project more and more people publish their data in the linked data cloud (Bizer, 2008). Moreover, the new web should be completely machine-understandable (Arup, 2011).

Furthermore, the growth of well-represented linked data cloud motivates researchers to improve web search functionality and exploit the machine understandable format.

On the other hand, one of the most important tools of the Internet is web search. Almost all Internet users use web search to find their needs. Consequently, improving searching RDF data is an urgent task. One challenge of searching web of data is getting implicit answers. In other words, search engines should go behind the raw data to understand web data and get query answers. Moreover, there is urgent need for smart search techniques that make the use of the new evolution of linked data.

The main objective of converting web data to RDF is to enable web agent to understand this data. Linked data is a web of machine understandable data (Correndo, 2010). However, web agents, that query linked data, cannot deeply understand RDF data. For example, consider a SPARQL query to get information about authors who are interested in *semantic data representation* from the corpus of semantic web conferences, http://data.semanticweb.org, which contains information about some conferences in semantic web filed. Although, the corpus contains the needed information, the user may obtain no results. This is because a query engine cannot get the implicit answers.

There are a lot of woke in searching RDF data. Many of these research based on SPARQL query language (Shady, 2010) (Olaf, 2009). SPARQL is a query language for RDF. SPARQL is quite similar to SQL (Axel, 2007). Moreover, there are many SPARQL endpoints attached to linked data sets to facilitate querying RDF data sets. The user can submit SPARQL query to these endpoints and get the results via http.

On the other hand, the dataset admin who represents the original data into RDF faces a problem of selecting RDF vocabularies because there are many equivalent vocabularies available on the Internet. Moreover, the client, which queries the RDF dataset is restricted to use the same vocabularies as RDF dataset.

This paper proposes creating a wrapper for SPARQL endpoint in which dataset admin adds a set of rules depending on the meaning of the dataset and the common asked queries. The proposed wrapper uses these rules to improve endpoint functionality.

Moreover, a related approach, which tries to express rules and infer additional RDF data, is SPIN (Holger, 2011). SPIN is a group of RDF properties that can be used to express rules. These rules

Farouk M. and Ishizuka M..
A RULE-BASED SPARQL ENDPOINT WRAPPER.
DOI: 10.5220/0003941407290732
In *Proceedings of the 8th International Conference on Web Information Systems and Technologies* (WEBIST-2012), pages 729-732
ISBN: 978-989-8565-08-2

attached to a specific ontology class and can be applied to infer data, or modify the current data. *spin:rule* property can be used to defined an inference rule using SPARQL construct or insert/delete.

Moreover, SPIN, which submitted to w3c to be discussed, adds rules to ontology level. However, our approach separates between rules level and ontology level. Separation between ontology and rules levels gives the user flexibility to add rules. In other words, it is difficult for the user to update the standard shared ontology to add his rules. Moreover, there are many users may add rules to infer the same property depending on their own data. The user wants to extend his data depending on the semantics of the data and the expected queries to be asked. Therefore, the users have different data want to make many rules even for the same ontology. Attaching rules to dataset gives flexibility to the users and avoids rules conflicts on ontology level.

As an example that will be used throughout this paper, consider that we will start a research project in semantic web, in which a huge amount of data should be represented into a semantic format. Some experiments will be done on this data to study the future of semantic web. It is decided that Tim Berner-Lee is the general manager for this project. The smart web agent, which is responsible for nominating project members, searches the linked data could to find qualified researchers. The agent may ask queries such as: Who are interested in semantic web, who are interested in semantic web and know Tim Berner-Lee, and so on.

The remainder of this paper is organized as follows. Section 2 describes the overall system architecture. Section 3 explains the idea of adding user-defined rules. Section 4 describes the proposed wrapper for SPARQL endpoint that exploits the defined rules. The experiments and results are discussed in section 5. Finally, section 6 provides the conclusion of this research.

2 SYSTEM ARCHITECTURE

The proposed wrapper consists of a set of user-defined rules and a simple inference engine to expand SPARQL queries based on the defined rules. Backward chaining is used to expand the queries. The wrapper sends SPARQL queries after expansion to the original endpoint. In addition, the wrapper collects the results of these queries and sends them back to the client. Figure 1 shows a general overview of the proposed wrapper.

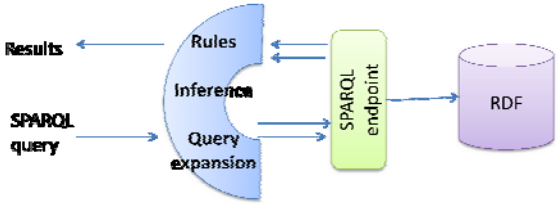

Figure 1: System overview.

Moreover, the query expansion process starts with finding appropriate rules in the defined rule set. Furthermore, rules index is used to facilitate finding rules during searching process. A matching between the retrieved rules and the user query produces expanded queries after replacing query conditions with rules premises.

3 ADDING RULES

The data publisher may add extra knowledge (rules), which considered as an extension for the original data. Moreover, adding this kind of knowledge enables the data admin, who selects specific vocabularies to represent the published data into RDF, to suggest alternatives vocabularies and expresses them in rules instead of duplicate RDF data with different vocabularies. In addition, the SPARQL endpoint admin should add a set of rules depending on the meaning of RDF dataset and common queries that users used to ask.

Moreover, there are two types of rules that can be added to the proposed wrapper. The first type focuses on discovering new RDF triples to get the implicit data, for example, the rules that infers the relation between topics and authors. This rules defined as: *if a person X is an author for the paper Y and the topic of the paper Y is T then the person X is interested in the topic T*. The other type of rules focuses on vocabularies mapping to allow client to use some different equivalent vocabularies. This will not restrict the user to use the same vocabulary as RDF dataset.

The format of user-defined rule is a simple production rule, "condition → action". The condition part syntax is the same as SPARQL query condition syntax. The action part is also represented into SPARQL syntax. Figure 2 shows an example for the rule: If a person A is an author to a paper Y, and a person B is an author to the paper Y, then → A knows B. The first part of the rule format is xml namespaces for the used vocabularies. The second part is the condition of the rule. The last part is the action part.

```
<rule id="2" // namespaces are omitted>
   <text>
     if two people are authors for the same paper, then
they know each others.
   </text>
<condition>
<con>
{
  ?ppr rdf:type iswc:InProceedings.
  ?person rdf:type foaf:Person.
  ?person2 rdf:type foaf:Person.
  ?ppr dc:Creator ?person.
  ?ppr dc:Creator ?person2.
  FILTER (?person != ?person2)
}
</con>
</condition>
<action>
{   ?person foaf:knows ?person2.  }
</action>
 </rule>
```

Figure 2: Example of user-defined rules.

Using these rules during the processing of original data to infer more data on the fly costs overhead processing time. However, using inference rules during searching give better results. Moreover, the proposed approach avoids data duplication and keeps the data size.

4 SEARCHING RDF

The proposed wrapper is an extension to the normal SPARQL endpoint. An inference step is added to the SPARQL endpoint to make the use of the user-defined rules. The proposed wrapper can answer some queries that cannot be answered using normal endpoint. Using inference, the engine goes behind the raw data to find implicit query answer.

The proposed wrapper is developed based on Jena SPARQL API with additional inference step. The actual usage of the added inference step is not to infer more data. However, the inference step is used for query expansion to get detailed queries that can be answered using the normal query engines.

4.1 Query Expansion

The proposed approach applies backward chaining to expand user query. The algorithm of SPARQL query expansion is a recursive algorithm that gets all possible queries based on a set of predefined rules. Indexing for the defined rules are established to link different rules based on rules action. This index for the defined rules facilitates finding the appropriate rules to expand a SPARQL query. The basic idea of this query expansion is to replace query condition with other conditions based on backward chaining of the rules.

Query expansion based on backward chining algorithm is as follows:

Input: SPARQL query, a set of rules

Output: a list of new queries equivalent to the inputted query

1. Get list of properties (predicates) used in query conditions.
2. Get related rules that can be used to expand the inputted query (based on rule index)
3. For each related rule
   - Match between rule actions and query conditions
   - Bind matched variables and keep them in a mapping state
   - Replace the matched query conditions with rule premises
   - Recursive call to expand the new query // *this call starts matching the new query and only the rest of rule set*
4. Combine all new queries in one list

The resulted queries are executed using the normal SPARQL endpoint. The results of these queries are combined and sent back to the client.

5 EXPERIMENT

In order to show the effectiveness of the proposed approach, we developed a web based application using JSP as a wrapper for semantic web conferences corpus endpoint, data.semanticweb.org. This data set contains information about some conferences related to semantic web field and some related information such as papers, authors, and so on. It contains around 180445 unique RDF triples. The developed wrapper is available on the Internet: http://cic012.cic.ci.i.u-tokyo.ac.jp:8080/wrapper/.

In this experiment, the developed wrapper uses a set of five rules. Example of these rules is:

- If a person A is an author to a paper Y, and a person B is an author to the same paper Y → A knows B.
- If a person A is an author to a paper Y, and the main topic of Y is T then → A is interested in T.
- If a topic X is a keyword for a paper Y → topic X is a subject of Y.

The first two rules infer implicit data. However, the third rule maps ontology vocabularies to facilitate finding query answer.

Table 1 shows the list of queries used in this

experiment. A set of queries are run using SPARQL endpoint of the semantic web conferences corpus, http://data.semanticweb.org/snorql/, and using the proposed wrapper. The results of this experiment are shown in table 2.

Table 1: List of Queries used in the experiment.

Number	Query
Q1	Who is interested in *Semantic Web*
Q2	Who knows *Jure Leskovec*
Q3	Who knows *Jure Leskovec* and is interested in *'social networks'*
Q4	Get papers in *semantic web* field
Q5	Get all papers of the author 'Gang Wang'
Q6	Get all papers in Ontology Learning field

The first query in table 1 asks about a person who is interested in semantic web field. The SAPRQL syntax for this query is as follow:

```
PREFIX foaf: http://xmlns.com/foaf/0.1/
select distinct ?ResearcherName where
{
?x foaf:name ?ResearcherName.
?x foaf:topic_interest 'Semantic Web'.
}
```

The RDF dataset does not contain topic_interest relation between authors and topics. Therefore the original endpoint returns no results. However, using inference rules the proposed wrapper expand the client query to another query depending on the defined rules set. Finally, the proposed wrapper can suggest 284 answers for this query.

In table 2, the third column shows the number of extra queries that generated by the wrapper. This number highly affects execution time.

Table 2: Query result using normal technique and the proposed approach.

Query number	Original endpoint	Proposed wrapper		
	Number of results	# of generated queries	Total number of results	Execution time
Q1	0	1	284	2.010
Q2	0	1	9	1.132
Q3	0	3	4	2.248
Q4	106	1	108	1.908
Q5	3	1	6	1.116
Q6	1	1	6	1.106

Numbers of the retrieved answers for some queries, such as Q4 and Q6, are increased because of using inference to get the equivalent vocabularies. Finally, using the proposed wrapper improves query results. It enables web agent not only to get more answers for queries but also to get answers for some queries cannot be answered using the original endpoint.

6 CONCLUSIONS

SPARQL endpoints facilitate querying RDF datasets over http. However, there are many queries cannot be answered. This paper proposes adding a new layer to the existing SPARQL endpoints as a wrapper. This wrapper facilitates writing queries for clients and increases the query answering recall. The proposed wrapper uses inference rules to go behind the raw data and get query answer based on a set of predefined rules. The experiments show that the wrapper can answer some queries that cannot be answered using the original endpoint. In addition, the proposed wrapper can used in vocabulary mapping to facilitate query writing for web clients.

REFERENCES

Axel Polleres, From SPARQL to rules (and back), Proceedings of the 16th international conference on World Wide Web (WWW2007), May 2007, Banff, Canada pages 787–796

Christian Bizer and Andreas Schultz. Benchmarking the Performance of Storage Systems that expose SPARQL Endpoints. *In Proceedings of the 4th International Workshop on Scalable Semantic Web knowledge Base Systems* (SSWS), 2008.

Arup Sarkar, Ujjal Marjit, Utpal Biswas “linked data generation for the university data from legacy database” *International Journal of Web & Semantic Technology* Year: 2011 Vol: 2 Issue: 3 Pages/record No.: 21-31

Shady Elbassuoni, M. Ramanath, R. Schenkel, and G. Weikum. Searching rdf graphs with SPARQL and keywords. *IEEE Data Engineering Bulletin*, 33(1), 2010

Correndo, G., Salvadores, M., Millard, I., Glaser, H., Shadbolt, N.: Sparql query rewriting for implementing data integration over linked data. *In: 1st International Workshop on Data Semantics* (2010)

Olaf Hartig, Christian Bizer, and Johann-Christoph Freytag: Executing SPARQL Queries over the Web of Linked Data. *In Proceedings of the 8th International Semantic Web Conference* (ISWC), Washington, DC, USA, Oct. 2009

Holger Knublauch, James A. Hendler, Kingsley Idehen “SPIN - Overview and Motivation”, http://www.w3.org/Submission/2011/SUBM-spin-overview-20110222/, February 2011.