Optimizing SPARQL Queries using Graph Traversal Algorithm
Jaykishan B. Likhya¹ Prof. Daxa V. Vekariya²
¹²Department of Computer Engineering
¹²Nobel Group of Institution-Junagadh, Gujarat, India

Abstract— Large amounts of interlinked semantic data are available for the semantic web. Linked open data are increasing exponentially on the web. This enables users to retrieve quality and complex information from the web, which include searching and querying linked data. World Wide Web Consortium has standardized SPARQL as a query language for the semantic linked open data. Current SPARQL query processing is still on its early stages for both scalability and efficiency. There are several approaches have been proposed but only few are implemented. This approach proposes the use of graph traversal algorithms for the optimization of SPARQL query. SPARQL query can be optimized using this approach in two phases. In first phase, we proposed to generate a query execution plan using graph traversal algorithm. Second phase focuses on to reduce the search space and optimize. To optimize the query execution, individuals which are not connected to any other individual are removed with acceptable loss of knowledge base. It represents a generic representation of graph for any ontology. The query is executed again on the revised graph which will have reduced query execution time with acceptable loss of knowledge base.

Key words: Semantic web, linked data, RDF, SPARQL

I. INTRODUCTION
The Web is increasingly understood as a global information space consisting not just of linked documents, but also of Linked Data. Tim Berners-Lee in May 2001 described his vision of a “new form of web content that is meaningful to Computers” and would “unleash a revolution of new possibilities” in an article called The Semantic Web. Traditionally, the data published on the web was made available as raw dumps in various formats. In the conventional Web the nature of relation between two web documents is implicit. But this Hypertext web i.e. HTML is not expressive enough to enable individual entities to be connected by typed links to related entities [14].

Linked Data is simply about using the Web to create typed links between data from different sources. These may be as diverse as databases maintained by two organizations in different geographical locations, or simply heterogeneous systems within one organization that, historically, have not easily interopereated at the data level. Technically, Linked Data refers to data published on the Web in such a way that it is machine-readable, its meaning is explicitly defined, it is linked to other external data sets, and can in turn be linked to from external data sets.

While the primary units of the hypertext Web are HTML (HyperText Markup Language) documents connected by untyped hyperlinks, Linked Data relies on documents containing data in RDF (Resource Description Framework) format (Klyne and Carroll, 2004). However, rather than simply connecting these documents, Linked Data uses RDF to make typed statements that link arbitrary things in the world.

One research area of the linked open data project is querying the linked data. Querying this linked data efficiently is essential with the growth of the linked data cloud. Linked data is always in the form of Resource Description Framework (RDF) format. After the development of different prototypes for querying this cloud structured data SPARQL protocol and RDF query Language was developed and accepted by the World Wide Web consortium as the query language to query Linked data. After this the major research area was to make efficient and fast queries, i.e. optimize the performance of the queries.

This optimization is required due to the growth of the data cloud which will grow even more eventually with more websites converging towards the Semantic Web.

A. Semantic Web:
The Semantic Web is a Web of data. There is a lot of data used every day, and it's not part of the Web. For example bank statements cannot be seen in calendar lines because we don't have a web of data. Data is controlled by applications, and each application keeps it to itself [16].

The vision of the Semantic Web is to extend principles of the Web from documents to data. Data should be accessed using the general Web architecture using, e.g., URI-s; data should be related to one another just as documents are This also means creation of a common framework that allows data to be shared and reused across application, enterprise, and community boundaries, to be processed automatically by tools as well as manually, including revealing possible new relationships among pieces of data.

Semantic Web technologies can be used in a variety of application areas; for example: in data integration, whereby data in various locations and various formats can be integrated in one, seamless application; in resource discovery and classification to provide better, domain specific search engine capabilities; in cataloguing for describing the content and content relationships available at a particular Web site, page, or digital library; by intelligent software agents to facilitate knowledge sharing and exchange; in content rating; in describing collections of pages that represent a single logical “document”; for describing intellectual property rights of Web pages and in many others.

B. Linked Data:
Linked Data is about using the Web to connect related data that wasn't previously linked, or using the Web to lower the barriers to linking data currently linked using other methods. More specifically, Linked Data is a term used to describe a recommended best practice for exposing, sharing, and connecting pieces of data, information,
and knowledge on the Semantic Web using URIs and RDF [19].

Key technologies that support Linked Data are URIs (a generic means to identify entities or concepts in the world), HTTP (a simple yet universal mechanism for retrieving resources, or descriptions of resources), and RDF (a generic graph-based data model with which to structure and link data that describes things in the world).

C. Resource Description Framework:
The Resource Description Framework (RDF) is a language for representing information about resources in the World Wide Web. It is particularly intended for representing metadata about Web resources, such as the title, author, and modification date of a Web page, copyright and licensing information about a Web document, or the availability schedule for some shared resource. However, by generalizing the concept of a “Web resource”, RDF can also be used to represent information about things that can be identified on the Web, even when they cannot be directly retrieved on the Web. [18]

RDF is intended for situations in which this information needs to be processed by applications, rather than being only displayed to people. RDF provides a common framework for expressing this information so it can be exchanged between applications without loss of meaning. Since it is a common framework, application designers can leverage the availability of common RDF parsers and processing tools. The ability to exchange information between different applications means that the information may be made available to applications other than those for which it was originally created. [18]

RDF is based on the idea of identifying things using Web identifiers (called Uniform Resource Identifiers, or URIs), and describing resources in terms of simple properties and property values. This enables RDF to represent simple statements about resources as a graph of nodes and arcs representing the resources, and their properties and values. The group of statements “there is a Person identified by http://www.w3.org/People/BL/contact#me, whose name is Berner Lee, whose email address is bl@w3.org, and whose title is Dr.” could be represented as the RDF graph in Figure 2.1

![RDF Graph](image)

Fig. 1: Sample RDF graph for Berner lee’s contact information

RDF also provides an XML-based syntax (called RDF/XML) for recording and exchanging these graphs.

1) Example 2.1:

```xml
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:contact="http://www.w3.org/2000/10/swap/pim/contact#">
  <contact:Person rdf:about="http://www.w3.org/People/BL/contact#me">
    <contact:fullName>Berner Lee</contact:fullName>
    <contact:mailbox rdf:resource="mailto:bl@w3.org"/>
    <contact:personalTitle>Dr.</contact:personalTitle>
  </contact:Person>
</rdf:RDF>
```

Like HTML, this RDF/XML is machine processable and, using URIs, can link pieces of information across the Web. However, unlike conventional hypertext, RDF URIs can refer to any identifiable thing, including things that may not be directly retrievable on the Web. The result is that in addition to describing such things as Web pages, RDF can also describe cars, businesses, people, news events, etc. In addition, RDF properties themselves have URIs, to precisely identify the relationships that exist between the linked items.

RDF is based on the idea that the things being described have properties which have values, and that resources can be described by making statements that specify those properties and values. RDF uses a particular terminology for talking about the various parts of statements. Specifically, the part that identifies the thing the statement is about is called the subject. The part that identifies the property or characteristic of the subject that the statement specifies is called the predicate, and the part that identifies the value of that property is called the object. [18]

So, taking the English statement  
http://www.example.org/index.html has a creator whose value is Berner Lee

The RDF terms for the various parts of the statement are:
- The subject is the URI
- http://www.example.org/index.html
- The predicate is the word “creator”
- The object is the phrase "Berner Lee”

D. SPARQL Protocol and RDF Query Language:

SPARQL is a general term for both a protocol and a query language. SPARQL is a syntactically-SQL-like language for querying RDF graphs via pattern matching. The language’s features include basic conjunctive patterns, value filters, optional patterns, and pattern disjunction [16,17].

The SPARQL protocol is a method for remote invocation of SPARQL queries. It specifies a simple interface that can be supported via HTTP or SOAP that a client can use to issue SPARQL queries against some endpoint [16,17].

1) A SPARQL query comprises of [13]:

- # prefix declarations
- PREFIX foo: http://example.com/resources/
- ...
- # dataset definition
- FROM
- ...
- # result clause
Given that the predicate of each triple pattern is bound, there are four possible types of sub-queries:

Type 1: a triple pattern where the subject is bound and the object is a variable.
For example: KeanuReeves starsIn ?movie.

Type 2: a triple pattern where the subject is a variable and the object is bound
For example: ?movie directedBy AndyWachowski.

Type 3: a triple pattern where both the subject and object are bound.
Type 4: a triple pattern where both the subject and object are variables.
For example: ?movie releaseYear ?year.

In this paper, static query execution plan are generated using two algorithms: Edmonds’ algorithm and Prim’s algorithm. Parallelized algorithms can be used as future work.

2) Technique 2 [12]:
Authors of “Evaluating Graph Traversal Algorithms for Distributed SPARQL Query Optimization [12]” have also described the use of graph traversal algorithms for SPARQL query.

In this paper, authors have focused on reducing network flow, and therefore they have combined Prim’s algorithm and bind join together to achieve the purpose. But still, several issues exist in the proposed pseudo-code, such as the sequence of popping out new triple patterns, pruning filler edges and retrieving new binding will not produce the right result.

Evaluation results of DSP Engine which is compared with DARQ Engine:

<table>
<thead>
<tr>
<th></th>
<th>DARQ Engine</th>
<th>DSP(Prim’s Algorithm) Engine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall runtime (s)</td>
<td>606.273</td>
<td>41.826</td>
</tr>
<tr>
<td>Queries per Second</td>
<td>0.08</td>
<td>1.20</td>
</tr>
<tr>
<td>Min/Max query runtime (s)</td>
<td>6.7576/26.3826</td>
<td>0.0148/3.3272</td>
</tr>
<tr>
<td>Average result count</td>
<td>15.82</td>
<td>15.82</td>
</tr>
<tr>
<td>CPU usage (%)</td>
<td>15</td>
<td>9</td>
</tr>
<tr>
<td>Memory usage (MB)</td>
<td>20 – 60</td>
<td>9</td>
</tr>
<tr>
<td>Up/Down network flow (MB)</td>
<td>39.32/14.09</td>
<td>0.88/0.42</td>
</tr>
</tbody>
</table>

Table 3.1: Evaluation results of existing system

Drawback of this approach is that, the execution time may be unacceptable when executing a complex query having many triple patterns. Another challenge is from co-reference phenomenon mean one resource can be identified by multiple URIs.

In future work, it is possible to explore mechanisms that enable executing multiple triple patterns in parallel. Another is to optimize SPARQL by co-reference and parallel query execution.
II. CONCLUSION

There are several approaches have been proposed but only few are implemented. Several approaches of distributed SPARQL query optimization have been proposed, including query rewriting, selectivity-based triple pattern reordering, and general methods for SPARQL optimization. In spite of the approaches proposed above optimizing techniques for SPARQL query are still on its early stages for both scalability and efficiency. Need of this research focuses on that graph traversal minimum spanning tree algorithm has its own characteristics that can contribute to the SPARQL query optimizing techniques.

REFERENCES

A. Research Papers:


B. Online Sources:


C. Books: