Scalable SPARQL Querying of Large RDF Graphs

Jiewen Huang, Daniel J. Abadi and Kun Ren

Yale Database Group
RDF Gaining Popularity

• Encouraged by major search engines
  • Google
  • Yahoo!
• More data sets available in RDF
  • Governments
  • Research communities
Linked Data Movement
Scalable Processing

- Single-node RDF management systems are abundant
  - Sesame
  - Jena
  - RDF-3X
  - 3store
- Research in clustered RDF management is less significantly explored: *The focus of the talk*
RDF as Triples and a Graph

<table>
<thead>
<tr>
<th>subject</th>
<th>predicate</th>
<th>object</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lionel Messi</td>
<td>type</td>
<td>footballer</td>
</tr>
<tr>
<td>Lionel Messi</td>
<td>playsFor</td>
<td>FC Barcelona</td>
</tr>
<tr>
<td>Lionel Messi</td>
<td>born</td>
<td>Rosario</td>
</tr>
<tr>
<td>Lionel Messi</td>
<td>position</td>
<td>striker</td>
</tr>
<tr>
<td>Xavi</td>
<td>type</td>
<td>footballer</td>
</tr>
<tr>
<td>Xavi</td>
<td>playsFor</td>
<td>FC Barcelona</td>
</tr>
<tr>
<td>Xavi</td>
<td>born</td>
<td>Barcelona</td>
</tr>
<tr>
<td>Xavi</td>
<td>position</td>
<td>midfielder</td>
</tr>
<tr>
<td>FC Barcelona</td>
<td>region</td>
<td>Barcelona</td>
</tr>
<tr>
<td>Barcelona, Spain</td>
<td>population</td>
<td>5,500,000</td>
</tr>
<tr>
<td>Josep Guardiola</td>
<td>manages</td>
<td>FC Barcelona</td>
</tr>
</tbody>
</table>
SPARQL

- RDF query language
- A basic graph pattern
- Answering SPARQL can be seen as finding subgraphs in the RDF data that match the graph pattern
Example for Star Pattern

- Find the names of the strikers that play for FC Barcelona.

SELECT ?name

WHERE { ?player type footballer .
    ?player name ?name .
    ?player position striker .
    ?player playsFor FC_Barcelona . }
Another Example

- Find football players playing for clubs in a populous region where they were born.

```sparql
SELECT ?player ?club ?region
WHERE {
  ?player type footballer .
  ?player playsFor ?club .
  ?player born ?region .
  ?club region ?region .
  ?region population ?pop .
  FILTER (?pop > 2,000,000) }
```
Data Partitioning

- Hash vs Graph partitioning
  - Hash: Only efficient for star patterns
  - Graph: Taking advantage of graph model

- Edge vs Vertex partitioning
  - Edge: Natural but inefficient for query execution
  - Vertex: Superior for common graph patterns
Edge/Triple Placement

- Minimizing data shuffling/exchange
  - Allowing data overlap
- N-hop guarantee
  - The extent of data overlap
  - If a vertex is assigned to a machine, any vertex that is within n-hop of this vertex is also stored in this machine
Example for N-Hop Guarantee
Query Processing

- Query execution is more efficient in RDF-stores than in Hadoop
  - Pushing as much of the processing as possible into RDF-stores
  - Minimizing the number of Hadoop jobs
  - The larger the hop guarantee, the more work is done in RDF-stores
To Communicate, or not to Communicate

- Given a query and n-hop guarantee, is communication (Hadoop job) between nodes needed?
  - Choose the “center” of the query graph
  - Calculate the distance from the “center” to the furthest edge
  - If distance > n, communication is needed; not needed otherwise
Back to the Example

- Find football players playing for clubs in a populous region where he was born.

```
SELECT ?player ?club ?region
WHERE { ?player type footballer .
  ?player playsFor ?club .
  ?player born ?region .
  ?club region ?region .
  ?region population ?pop .
FILTER (?pop > 2,000,000) }
```
Experimental Setup

- 20-machine cluster
- Leigh University Benchmark (LUBM): 270 million triples
- Competitors:
  - Single-node RDF-3X
  - SHARD: triple-store system in Hadoop
  - Graph partitioning (the proposed system)
  - Hash partitioning on subjects
Performance Comparison

![Graph showing performance comparison between different partitioning methods. The y-axis represents wall-clock time in seconds on a logarithmic scale, and the x-axis represents LUBM queries. The methods compared include Single-node RDF-3X, SHARD, and Hash Partitioning, as well as Graph Partitioning (1-hop) and Graph Partitioning (2-hop).]
Speedup

- Better than linear speedup
Summary

- We propose a new architecture for scalable RDF data management: *RDF-stores + Hadoop*
- We propose a new approach for data placement and corresponding query processing: *Graph partitioning + N-hop guarantee*
- The techniques in the talk can be generalized to the problems of subgraph pattern matching in other graphs
- The lesson we learned: Inter-node communication is expensive, avoid it.
Thank you!
Backup Slides: Optimization

- Problem: High-degree vertexes make the graph well-connected and difficult to partition
- Solution: Removing them in graph partitioning

- Problem: High-degree vertexes cause data explosion in n-hop guarantee
- Solution: Weakened n-hop guarantee