Scalable and efficient SPARQL querying over Distributed RDF Data

Mebin Jacob
CISE, University of Florida

BACKGROUND

• RDF, stands for Resource Description Framework, is a family of World Wide Consortium (W3C).
• Since it’s schema free, it makes RDF a flexible mechanism for describing entities in a way that many different data publishers (located across the internet) can add arbitrary information about the same entity or create links between disparate entities.
• SPARQL queries, is an RDF query language a semantic query language for databases, able to retrieve and manipulate data stored in Resource Description Framework (RDF) format.
• Factor graphs, a factor graph is a hyper graph with vertices as facts, and factors (rules) as hyper edges connecting the vertices.
• Grounding is the process of writing the graph to disk so that it can be used to perform inference.

PURPOSE

1. To analyze and benchmark RDF data stores, Jena SDB and RDF-3x using LUBM and DBPedia dataset. This is done to analyze the performance of various kinds of queries over the distributed setup which used n-hop guarantee.
2. The graph partitioning with n-hop guarantee is a natural fit for k-hop network approximation in marginal inference over factor graph. Thus we extend the DSPARQ architecture to support inference over factor graph and provide the partitioning analysis.
3. Analysing factor graphs for graph partitioning and n-hop guarantee, that is the feasibility of graph partitioning and n-hop guarantee over factor graphs.

MATERIALS AND METHODS

Implementation Stack
1. We use six node cluster for our benchmark purpose.
2. All six node cluster are implemented over docker.
3. Five nodes are configured with rdf3x and Jena sdb, with java/scala binaries and hadoop slave instances configured.
4. One node has hadoop master configured along with java/scala binaries.
5. Everything’s within Docker! hence portable.

Graph Partitioning
We use multilevel k-way partitioning scheme to partition out graph data into roughly equal sizes. We use METIS to achieve this partitioning scheme.

N-Hop Guarantee
We implement an algorithm to ensure that for any given node in the original partition we can get all nodes that are at most n-hop away, where n may take the values 1, 2, 3 and so on.

Note :- This technique is known to work well only on sparse graphs.

This same technique is implemented in python and analysis is done over factor graphs.

RESULTS

• Benchmark of various LUBM queries over RDF-3X engine.

CONCLUSIONS

• The k-hop approximate inference can be easily parallelized on each node, the throughput can be easily increased linearly w.r.t to the number of nodes.
• The overlap ratio is high for factor graphs but acceptable.

REFERENCES


Thanks to Dr. Wang, Dr. Milenko and Xiaofeng.