Distributed Big RDF Data Processing and Data Partitioning using hadoop in Cloud

Tejas Bharat Thorat
Dept. of Computer Engineering
MIT Academy of Engineering,
Alandi, Pune, India

Prof.Ranjana R.Badre
Dept. of Computer Engineering
MIT, Academy of Engineering
Alandi, Pune, India

Abstract—Big data business can leverage and bene?t from the Clouds. The processing and partitioning of big data in cloud is an important challenge. In this system big RDF datasets is considered as focus for the investigation. The open data movement initiated by the W3C publishes data in RDF format on the Web. By 2011, the Linked Open Data (LOD) project had published 295 datasets consisting of over 31 billion RDF triples, interlinked by around 504 million RDF link [10].Such huge RDF graphs can easily reduce the performance of any single server due to the limited CPU and memory capacity. The proposed system will use data partitioning framework called Hadoop, for processing of distributed big RDF graph data. This system uses the Graph Signature, approach for indexing RDF graphs which will improve the efficiency of query processing.

Index Terms—Query processing, RDF graph data, Apache Hadoop, SPARQL, Cloud, Graph Signature.

I. INTRODUCTION

For efficient big data processing, Cloud computing infrastructures are widely recognized as an attractive computing platform because it minimizes the cost for the large-scale computing infrastructure. With Linking Open Data community project and World Wide Web Consortium (W3C) advocating RDF (Resource Description Framework) [11] as a standard data model for Web resources, it has been witnessed a steady growth of both big RDF datasets and large and growing number of domains and applications capturing their data in RDF and performing big data analytics over big RDF datasets. Recently the UK (United Kingdom) government is publishing RDF about its legislation [3] with a SPARQL (a standard query language for RDF) query interface for its data sources [4]. More than 52 billion RDF triples are published as of March 2012 on Linked Data [5] and about 6.46 billion triples are provided by the Data gov Wiki [6] as of February 2013.

A. RDF and SPARQL

An RDF dataset consists of (subject, predicate, object) triples (or so-called SPO triples) with the predicate representing a relationship between its subject and object. An RDF dataset can be depicted as an RDF graph with subjects and objects as vertices and predicates as labeled edges connecting a pair of subject and object. Each edge is directed, emanating from its subject vertex to its object vertex. Fig. 1(a) shows an example RDF graph based on the structure of SP2Bench(SPARQL Performance Benchmark)[7].

SPARQL is a SQL-like standard query language for RDF recommended by W3C. Fig. 1(b) shows an example of SPARQL Queries. Most SPARQL queries consist of multiple triple patterns, which are similar to RDF triples except that in each triple pattern, the subject, predicate, object can be a variable.

Hadoop MapReduce programming model and Hadoop Distributed File System (HDFS) are one of the most popular distributed computing technologies for distributing big data processing across a large cluster of compute nodes in the Cloud. However, processing the huge RDF data using Hadoop MapReduce and HDFS poses a number of new technical challenges. When viewing a big RDF dataset as an RDF graph, it typically consists of millions of vertices (subjects or objects of RDF triples) connected by millions or billions of edges (predicates of RDF triples). Thus triples are correlated and connected in many different ways. The triples Graph is created according to the variable. The variable might be subject or object. Data partitions generated by such a simple partitioning method tend to have high correlation with one another. Thus, most of the RDF queries need to be processed through multiple rounds of data shipping across partitions hosted in multiple compute nodes in the Cloud. HDFS is excellent for managing big table like data where row objects are independent and thus big data can be simply divided into equal-sized chunks which can be stored in a distributed manner and processed in parallel and reliably. However, HDFS is not optimized for processing big RDF datasets of high correlation. Therefore, even simple retrieval queries can be quite inefficient to run on HDFS. However, Hadoop MapReduce programming model is optimized for batch-oriented processing jobs over

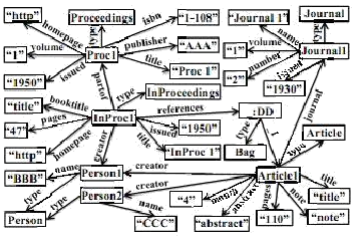

Fig. 1. Example of RDF Graph

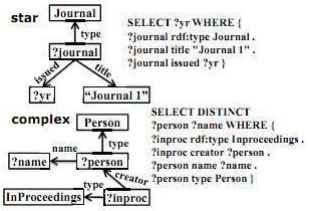

Fig. 2. Example of SPARQL

big data rather than real-time request and respond types of jobs. Thus, without correlation preserving data partitioning, HadoopMapReducealone is neither adequate for handling RDF queries nor suitable for structure-based reasoning on RDF graphs. Considering these challenges, a Scalable and customizable data Partitioning for processing of distributed big RDF graph data system has been proposed which helps for fast data retrieval. It allows efficient and customizable data partitioning of large heterogeneous RDF graphs by using Graph Signature. It supports fast processing of distributed big data of various sizes and complexity. This system has the design adaptive to different data processing demands in terms of structural correlations. The system proposes a selection of scalable parallel processing techniques to distribute the structured graph among the computing nodes in the cloud in such a manner that it minimizes inter-node communication cost. The cost is minimized by localizing most of the queries to independent partitions and by maximizing intra-node processing. The partitioning and distributing of big RDF data using Graph signature, system can considerably reduce the inter node communication cost of complex query processing as most RDF queries can be evaluated locally on a partition server. There is no need of data shipping from other partition nodes.

II. RELATED WORK

Graph Partitioning has been extensively studied in several communities for several decades [8], [9]. Recently a number of techniques have been proposed to process RDF Queries on a cluster machine.

A. Efficient and Customizable data Partitioning Framework for Distributed Big RDF Data Processing in the Cloud [1]

In this system SPA (Scalable and yet customizable data Partitioning framework) is proposed, for processing of distributed big RDF graph data in the cloud. [1]. A scalable parallel processing techniques to distribute the partition blocks across a cluster of compute nodes is been developed in such a manner that it reduces the inter-node communication cost by localizing most of the queries on distributed partitions.

1) Limitation: In this system a data partitioning is done on a vertex-centric approach. This Vector blocked based partitioning techniques is not much efficient for Large RDF Graph data.

B. Efficient SPARQL Query Evaluation In a Database Cluster [2]

This System presents the data partitioning and placement strategies, as well as the SPARQL query evaluation and optimization techniques in a cluster environment. In this system RDF data partitioning solution is provided by logically partitioning the RDF triples based on the schemas . By using these logical partitions different query evaluation tasks are assigned to different logical partitions.

1) Limitation: In this system partitioning is schema based.

C. Other Existing Systems

Storing, Indexing and Querying Large Provenance Data Sets as RDF Graphs in Apache HBase[12], in this paper the problem of storing and querying large collections of scientific workflow provenance graphs serialized as RDF graphs in Apache HBase is studied. The algorithms used rely on indices to compute expensive join operations, make use of numeric values that represent triple positions rather than actual triples with lengthy literals and URIs, and eliminate the need for intermediate data transfers over a network.

III. METHODOLOGY FOR PROPOSED SYSTEM

In the proposed system the data used is in RDF format. This system can be summarized as a) The partitioning of

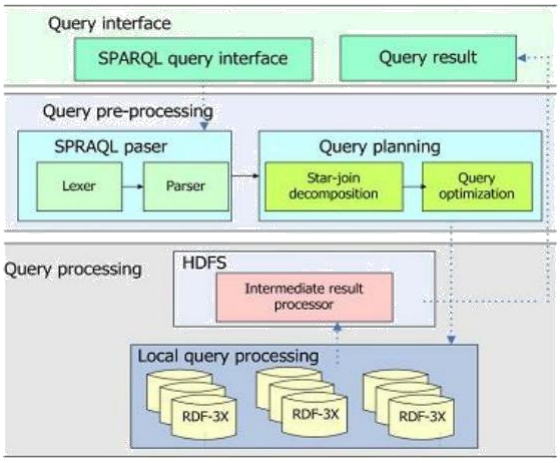

Fig. 3. The Framework Of Query Engine

RDF data files. b) The fast data retrieval by efficient query execution of SPARQL query.

The first prototype of the system is implemented on top of Hadoop Map Reduce and HDFS. This system consists of one coordinator and a set of worker nodes (VMs) in the cluster. The coordinator serves as the Name Node of HDFS and the Job Tracker of Hadoop Map Reduce and each worker serves as the Data Node of HDFS and the Task Tracker of Hadoop Map Reduce. To efficiently store the generated partitions RDF-specific storage system is used on each worker node.

A. Partitioning of Big RDF data

The RDF data file is distributed on the VM's. The file distribution is done on the bases of the nodes. Each graph in the Graph Signature is a directed labeled graph, it is easy to store them in the same way the data graph is stored. In this system a data graph and its O/I-signatures will be stored and queried in the same way, using the Oracle Semantic Technology. To store the O/I-signature's extent, every node in SO and SI is given an identifier. This id is used in a lookup table to store the nodes. This table is called the graph signature extent. For query formulation, we only store node labels and their group ids in a table as specified above.

B. Fast data Retrieval by efficient query execution

The Definition of the Graph Signature Definition(Graph Signature). Given an RDF graph G, its Graph Signature S is comprised of two summaries: O-Signature So and I-Signature S_i. In short, S=<So, S_i>. Definition (O-Signature). Given an RDF graph G, its So is a directed labeled graph, such that, each node in So is an equivalent class (=o) of some nodes in G; and each edge p in So from u to v (u^p– > v) iff G contains an edge p from a to b (a^p–>b) and a€ u, b€ v. Definition (I-Signature). Given an RDF graph G, its S_i is a directed labeled graph, such that, each node in S_i is an equivalent class (=I) of some nodes in G; and each edge p in S_i from u to v (u^p -> v) iff G contains an edge p from a to b(a^p–>b) and a € u, b € v. To compute the Graph Signature the standard algorithm for computing bisimilarity [13] will be used. This system will be using the modified algorithm to suit RDF Graph, for computing both the O-signature and I-signature. The input in each algorithm is a data graph and the output is O/I signature.

IV. CONCLUSION

In this proposed system, an efficient and customizable RDF data partitioning model will be used for processing of distributed big RDF graph data in the Cloud. The main contribution is to develop distributed partition of RDF data across a cluster of compute nodes in such a manner that minimizes the inter-node communication cost. This system shows that the system will be efficient for partitioning of big RDF datasets of various sizes and structure but also will be effective for fast processing RDF queries of different types of complexity using Graph Signature method.

ACKNOWLEDGMENT

I express true sense of gratitude towards my project guide Prof. Ranjana R. Badre, of computer department for her invaluable co-operation and guidance that she gave me throughout my research. I specially thank our P.G coordinator Prof. R. M. Goudar for inspiring me and providing me all the lab facilities, which made this research work very convenient and easy. I would also like to express my appreciation and thanks to our HOD Prof. Uma Nagaraj and Principal Dr. Y. J. Bhalerao and all my friends who knowingly or unknowingly have assisted me throughout my hard work.

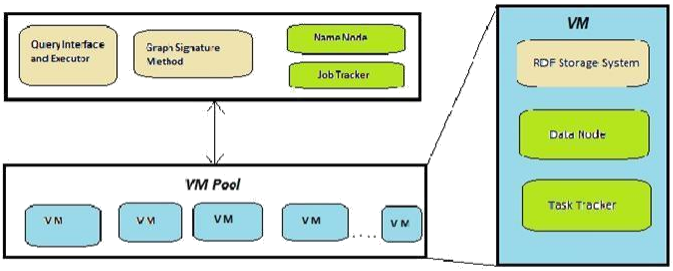

Fig. 4. The architecture of proposed system

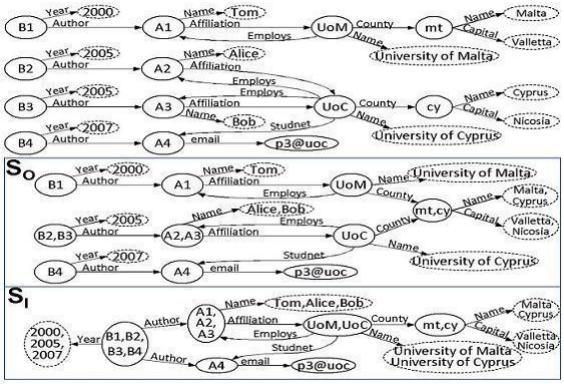

Fig. 5. RDF Data Graph and its I/O Signatures

REFERENCES

[1] Kisung Lee, Ling Liu, Yuzhe Tang, Qi Zhang, Yang Zhou Ef?cient and Customizable Data Partitioning Framework for Distributed Big RDF Data Processing in the Cloud,2013 IEEE Sixth International Conference on Cloud Computing

[2] Fang Du1,2, Haoqiong Bian1, Yueguo Chen1, Xiaoyong Du1dfang,bianhq,chenyueguo,duyong@ruc.edu.cn Ef?cient SPARQL Query Evaluation In a Database Cluster, 2013 IEEE International Congress on Big Data.

[3] http://www.legislation.gov.uk/developer/formats/rdf.

[4] http://data.gov.uk/sparql

[5] T. Heath, C. Bizer, and J. Hendler, Linked Data, 1st ed., 2011

[6] http://data-gov.tw.rpi.edu/wiki

[7] M. Schmidt, T. Hornung, G. Lausen, and C. Pinkel, SP2Bench: A SPARQL Performance Benchmark,in ICDE, 2009.

[8] B. Hendrickson and R. Leland,A multilevel algorithm for partitioninggraphs, in Supercomputing, 1995

[9] G. Karypis and V. Kumar,A Fast and High Quality Multilevel Schemefor Partitioning Irregular Graphs,SIAM J. Sci. Comput., 1998.

[10] http://www.w3.org/wiki/SweoIG/TaskForces/CommunityProjects/LinkingOpenData

[11] http://www.w3.org/RDF

[12] ArtemChebotko, John Abraham, Pearl Brazier, Anthony Piazza, AndreyKashlev, Shiyong Lu ,Storing, Indexing and Querying Large Provenance Data Sets as RDF Graphs in Apache HBase,2013 IEEE Ninth World Congress on Services.

[13] R. Paige and R. Tarjan,Three Partition Refinement Algorithms,SIAM J. Computing, vol. 16, no. 6, pp. 973-989, 1987.

[14] Mustafa Jarrar and Marios D. Dikaiakos, Member, IEEE Computer Society ,A Query Formulation Language for the Data Web ,IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 24, NO. 5, MAY 2012

[15] RuiWang , Kenneth Chiu ,A Stream Partitioning Approach to Processing Large Scale Distributed Graph Datasets,2013 IEEE International Conference on Big Data

[16] Z Craig Franke, Samuel Morin, ArtemChebotko , John Abraham, and Pearl Brazier ,Distributed SemanticWeb Data Management in HBase and MySQL Cluster,2011 IEEE 4th International Conference on Cloud Computing

[17] Fang Du1,2, Haoqiong Bian1, Yueguo Chen1, Xiaoyong Du1 ,Ef?cient SPARQL Query Evaluation In a Database Cluster,2013 IEEE International Congress on Big Data

[18] Resource Description Framework (RDF) http://www.w3.org/RDF/.

