Middle-East Journal of Scientific Research
24 (Recent Innovations in Engineering, Technology, Management & Applications): 13-17, 2016
ISSN 1990-9233; © IDOSI Publications, 2016
DOI: 10.5829/idosi.mejsr.2016.24.RIETMA103

An Efficient and Scalable Method for RDF Datasets Using Map Reduce Paradigm

B. Nagarajan, C. Santhosh Kumar and R. Dhandapani

Department of CSE, Priyadarshini Engineering College, Vaniyambadi, India

Abstract: An improved scheme for large volume Resource Distribution Framework (RDF) datasets using Map Reduce, for incremental knowledge base which produces high-performance reasoning and runtime searching. The centralized reasoning scheme is not able to process large datasets. With severe growth of semantic data and ontology bases has brought considerable challenges in performing well-organized and scalable reasoning. RDF is a basic representation of ontologies used to describe the knowledge in the Semantic Web. Improved distributed reasoning methods are required to progress the performance and scalability of inferences. We propose a new representation Transfer Interference Forest/Effective Assertional Triples to bear incremental inference over Resource Distribution Framework large-scale datasets. It can reduce the storage requirement and eases the reasoning process. An efficient and scalable reasoning method called Improved Interference Scheme is presented based on Transfer Interference Forest/Effective Assertion Triples and the consequent searching strategy is given to satisfy end-users online query needs.

Key words: MapReduce · Ontology Reasoning · RDF · Semantic Web

INTRODUCTION

There is different applications with a large volume of Semantic Web data [1] and its fast growth, have emerged such as expert systems, e-marketplace, Web services composition, cloud system management, healthcare and life sciences and etc.

The semantic web data [2] still growing in recent years, now it reached over 30 billion triples. RDF is a basic representation of ontologies used to describe the knowledge in the Semantic Web. Deriving inferences in the large-scale RDF files, referred to as large-scale reasoning, poses challenges in: the growing amount of information requires scalable computation capabilities for large datasets, distributed data on the web make it difficult to acquire appropriate triples for appropriate inferences; fast processing for inferences is required to satisfy the requirements of online query.

Due to the performance limitation of a centralized architecture executed on a single machine or local server when dealing with large datasets, distributed reasoning approaches executed on multiple computing nodes have thus emerged to improve the scalability and speed of inferences.

In this paper, we proposed an Improved Interference Scheme(IIS) for large-scale RDF datasets via MapReduce [3]. To store RDF datasets triples we introduce new concepts: Transfer Interference Forest (TIF) and Effective Assertion Triples (EAT). They can reduce the storage process and simplify the reasoning process. More ever, TIF and EAT requires minimum computation for datasets updation. Using Hadoop we implemented a prototype and it allows performing experiments on Billion Triples Challenge benchmark. Overall our system can efficiently reduce the storage requirements, eases the reasoning process and IIS produces the corresponding searching strategy to satisfy the end users' online query needs.

Related Works: Semantic web inference has fascinated much attention from both academics and industry nowadays. Anagnostopoulos and Hadjiefthymiades [4] proposed two fuzzy inference engines based on the knowledge-representation model to enhance the context inference and classification for the well-specified information in Semantic Web.

Paulheim and Bizer [5] proposed the problem of inference on noisy data and presented the SD Type method in RDF datasets to deal with noisy data. Guo et al. [6] introduced a new RuleXPM approach that consisted of a semantic inference engine on a multiphase forward-chaining algorithm to solve the semantic inference problem in heterogeneous e-marketplace activities.

Corresponding Author: B. Nagarajan, Department of CSE, Priyadarshini Engineering College, Vaniyambadi, India.

Hendler and Weaver [7] presented a method for materialize the complete finite RDF closure in a scalable manner and evaluated it on hundreds of millions of triples. Urbani [8] proposed a scalable distributed reasoning method for computing the closure of an RDF graph based on MapReduce and implemented it on top of Hadoop.

The storage of RDF closure is not a small amount and the query on it takes nontrivial time. Moreover, as the data volume increases and the ontology base is updated, these methods require the recompilation of the entire RDF closure every time when new data arrive. To avoid such time-consuming process, incremental reasoning methods are proposed in [9, 10, 11].

In this paper presents new a method IIS based on MapReduce and Hadoop, to speed up the updating process with newly-arrived data and fulfill the requirements of end-users for online queries, which can well influence the old and new data to reduce the updating time and decrease the reasoning time when facing large RDF datasets.

Preliminaries

Map Reduce: In this paper, we proposed Improved Interference Scheme(IIS) based MapReduce and Hadoop platform. MapReduce is a programming paradigm for distributed and parallel processing of batch jobs. Each jobs contains a map with a reduce, in which map assigns a key each element and partitions the input data, while reduce process each partition in parallel and merges all intermediate values with the same key into final results.

The MapReduce programming model provides an efficient way to process semantic sets of data [12]. MapReduce is the heart of Hadoop. It is this programming paradigm that allows for massive scalability across hundreds or thousands of servers in a Hadoop cluster. The MapReduce concept is fairly simple to understand for those who are familiar with clustered scale-out data processing solutions.

Hadoop [13] supports data-demanding distributed applications on huge clusters of MapReduce. Hadoop uses a distributed file system, data transfer execution management, error management and job scheduling and can run and monitor MapReduce applications on groups of commodity machines.

Hbase [13] is a distributed and scalable data store for Hadoop. It provides real-time read/write access to very large tables (billions of rows and millions of columns) on clusters of commodity hardware. Because of its features in linear scalability, automatic failover support and convenient backup of MapReduce jobs, uses it for real-time queries of RDF triples.

Resource Description Framework (PDF): Semantic Web is based on RDF, which integrate a range of applications by using XML for syntax and URI for naming. RDF is an assertion language proposed to be used to express propositions using specific formal vocabularies. RDF data model is like classic conceptual modeling approach, as it is based on the idea of making statements about resources.

RDF triple is used to describe the relationship between two things. Its formal definition is <subject, predicate, object> which denotes resource, properties and express the relationship between resource and object. RDF schema is a set class with certain properties. It provides essential elements for the account of ontologies, or called RDF vocabularies, planned to structure RDF resources. These resources can be saved in a triple store to reach them with the simple protocol and RDF query language.

Improved Interference Scheme(iis) over Large-scale Rdf Datasets: This chapter presents over IIS over large-scale RDF datasets. The figure.1 shows the overview and detailed steps of IIS.

The input of the system is incremental RDF data files. Then the dictionary encoding and triples indexing module encodes the input triples and for each triple an index is built based on an inverted index method. After that the incremental triples are separated into the incremental ontological triples and incremental assertion ones. The TIF/EAT construction Module generates the TIF and EAT based onthe ontological and assertional triples. For the second time and thereafter, the TIF/EAT Update Module only updates relative TIF and EAT.

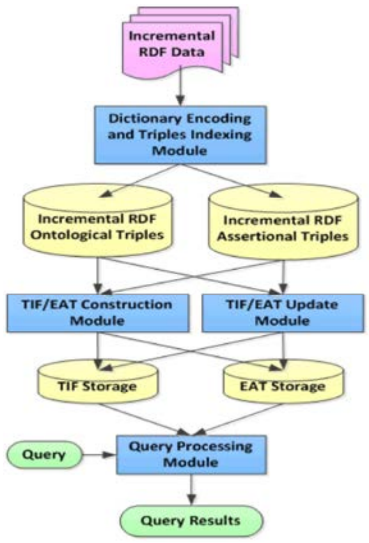
Incremental RDF Data
Dictionary Encoding and Triples Indexing Module
Incremental RDF Ontological Triples
Incremental RDF Assertional Triples
TIF/EAT Construction Module
TIF/EAT Update Module
TIF Storage
EAT Storage
Query
Query Processing Module
Query Results

Fig. 1: Steps in IIS

The created or updated ones are stored in TIF and EAT storages, respectively. The query processing module takes users' queries as input and reasons over the TIF and EAT to obtain the query results. Each module is introduced next.

Dictionary Encoding and Triples Indexing: The dictionary encoding and triples indexing module encodes all the triples into a unique and small identifier to reduce the physical size of input data. Then the ontological and assertional triples are extracted from the original RDF data. To efficiently compress a large amount of RDF data in parallel, we run a MapReduce algorithm on input datasets to scan all the URIs line by line and for each URI, a unique numeric ID is generated by the hash code method. The corresponding relationship between the original URI and its code is stored in a table named "Encode" in HBase.

TIF Construction: TIF is a set of directed trees as constructed by the triples whose predicates are rdfs: sub Class of, rdfs: sub Property of, rdfs: domain and rdfs: range.

It is further divided into Property TIF (PTIF), class TIF (CTIF) and domain/range transfer forest (DRTF).

The Construction of PTIF Includes Two Steps:

- Abstract all the triples whose predicates are rdfs:subPropertyOf and for each triple, build a directed graph from nodes A to B, in which A stands for its subject and B stands for its object.
- Abstract the triples whose predicate is rdf: type and object is rdfs: Container-Membership Property and following the 12th RDFS rule (prdf:type rdfs: Container Membership Property => p rdfs: sub Property of rdfs: member), build a directed graph from nodes A to B, in which A is its subject and B is rdfs:member.

The Construction of CTIF Contains Three Steps:

- Abstract all the triples whose predicates arerdfs:subClassOf and for each triple, build a directed graph from nodes A to B, in which A stands for its subject and B stands for its object.
- Abstract the triples whose predicate is rdf:type and object is rdfs:class and RDFS rule (srdf: typerdfs: class=>srdfs: sub Class of rdfs:resource), build a directed graph from nodes A to B, in which A is its subject and B is rdfs:Resource.
- Abstract the triples whose predicate is rdf:type and object is rdfs:Datatype and RDFS rule (ordf:type rdfs:Datatype => o rdfs:subClassOf rdfs:Literal), build a directed graph from nodes A to B, in which A is its subject and B is rdfs:Literal.

The Construction of DRTF Is as Follows:

- Abstract all the triples whose predicate is dfs:range and for each triple, build a directed graph from nodes A to B, in which A stands for its subject and B stands for its object.
- Abstract all the triples whose predicate is rdfs:domain and for each triple, build a directed graph from nodes A to B, in which A stands for its subject and B stands for its object. Note that the arrow connecting A and B is dotted line.

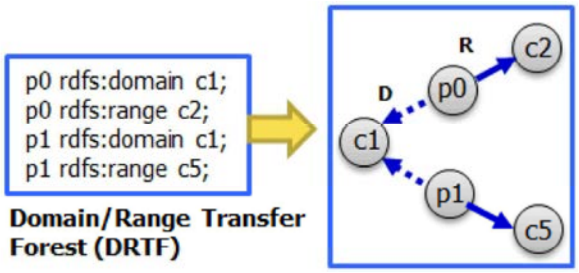

Fig. 2: DRTF construction

Reasoning over TIF

Algorithm 1

```
Reasoning Over PTIF
begin
for each node p in PTIF,
Q ← forward path of p
for each node q in Q,
add triple <s, q, o> to R // generate the derivedtriples
return R
end
```

Algorithm 2

```
Reasoning Over DRTF
begin
for each node p in DRTF,
if p has a domain edge linked to node c
add triple <s, rdf:type, c> to R
if p has a range edge linked to node c
add triple <o, rdf:type, c> to R
return R
end
```

```
Algorithm 3
Reasoning Over CTIF
begin
for each node o in CTIF,
C ← forward path of o
for each node c in C,
add triple <s, rdf:type, c> to R
return R
end
```

Incremental Update for TIF and EAT: The advantages of constructing TIF and EAT are twofold, reducing the storage space since we only store the core and minimum information that cannot be derived and more importantly, providing an efficient way for updating the knowledge base because updating TIF and EAT takes much fewer efforts than changing the entire ontology and recomputing RDF closure.

When new RDF files arrive, new edges are added to the existing TIF. Now we have two kinds of edges, i.e., existing edges referring to the triples that exist in the original TIF and incremental ones to those whose subject or object or both do not exist.

The steps for updating EAT are as follows

Step 1: Generate new TIF by adding incremental edges to the existing TIF.

Step 2: Generate incremental EAT based on the incremental ontological triples, add the incremental EAT to the existing EAT and run Algorithm 4 to calculate new EAT.

Step 3: Generate incremental DRTF based on the incremental ontological triples and add incremental DRTF to the existing DRTF.

Step 4: For the EAT with a predicate in the reverse path of the incremental edges while the forward path of the incremental edge contains nodes in DRTF, generate the assertional triples.

Step 5: Generate new TIF by adding incremental edges to the existing TIF.

Step 6: Generate the incremental EAT based on the incremental assertional triples and the triples generated in Step 4, add the incremental EAT to the existing EAT and run to calculate new EAT.

System Implementation and Comparison

System Architecture: To validate our proposed approaches, a prototype is implemented on the Hadoop platform that is widely used to enable the MapReduce technology.

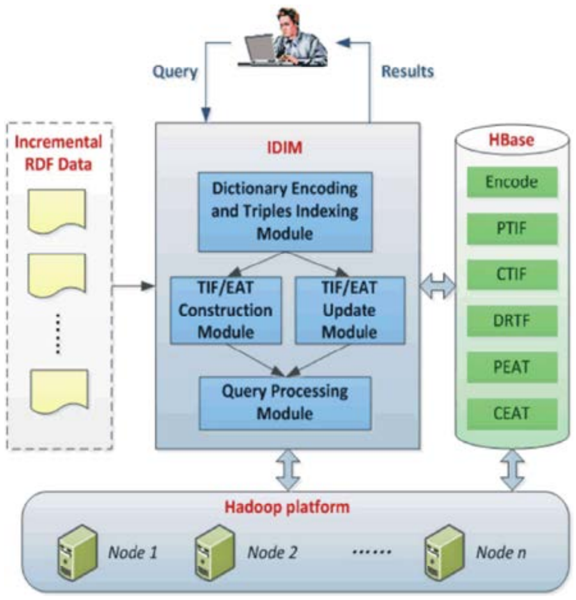

Fig. 3: Architecture

Figure 3 presents the architecture of the prototype system. The core of the system is the IDIM modules, which receive the input incremental RDF datasets, process the triples and perform the reasoning by a set of MapReduce programs, interact with Hbase for storing or reading the intermediate results and return the query results to end-users. We have designed six HBase tables to store the encoded ID, TIF, RTF, EAT and CAT. The Hadoop [13] framework is an opensource Java implementation of MapReduce that allows for the distributed processing of large datasets across clusters of computers via simple programming models. It can scale up from single servers to thousands of machines, each offering local computation and storage and manages execution details such as data transfer, job scheduling and error management.

Performance Evaluation: BTC consists of five large datasets, i.e., Datahub, DBpedia, Freebase, Rest and Timbl and each dataset contains several smaller ones. In order to show the performance of our method, we compare IDIM with WebPIE [9], which is the state-of-the-art for RDF reasoning. As the purpose of this paper is to speed up the query for users, we use WebPIE to generate the RDF closure and then search the related triples as the output for the query. The Hadoop configurations are identical to that in IDIM. Then the comparison can be concentrated on the difference of reasoning methods.

CONCLUSION

In the big data era, reasoning on a Web scale becomes increasingly challenging because of the large volume of data involved and the complexity of the task. Full reasoning over the entire dataset at every update is too time-consuming to be practical. This paper for the first time proposes an IIF to deal with large-scale incremental RDF datasets to our best knowledge. The construction of TIF and EAT significantly reduces the recomputation time for the incremental inference as well as the storage for RDF triples. Meanwhile, users can execute their query more efficiently without computing and searching over the entire RDF closure used in the prior work. Our method is implemented based on MapReduce and Hadoop by using a cluster of up to eight nodes. We have evaluated our system on the BTC benchmark and the results show that our method outperforms related ones in nearly all aspects.

In the future, we will validate our methods on more datasets and extend IDIM to OWL and other ontology languages.

REFERENCES

1. Kourtesis, D., J.M. Alvarez-Rodriguez and I. Paraskakis, 2014. "Semanticbased QoS management in cloud systems: Current status and future challenges," Future Gener. Comput. Syst., 32: 307-323.
2. Linking Open Data on the Semantic Web [Online].Available:http://www.w3.org/wiki/TaskForces/CommunityProjects/LinkingOpenData/DataSets/Statistics.
3. Dean, J. and S. Ghemawat, 2008. "MapReduce: Simplified data processing on large clusters," Commun. ACM, 51(1): 107-113.
4. Anagnostopoulos, C. and S. Hadjiefthymiades, 2009. "Advanced inference in situation-aware computing," IEEE Trans. Syst., Man, Cybern. A, Syst., Humans, 39(5): 1108-1115.
5. Paulheim, H. and C. Bizer, 2013. "Type inference on noisy RDF data," in Proc ISWC, Sydney, NSW, Australia, pp: 510-525.
6. Guo, J., L. Xu, Z. Gong, C.P. Che and S.S. Chaudhry, 2012. "Semantic inference on heterogeneous e-marketplace activities," IEEE Trans. Syst., Man, Cybern. A, Syst., Humans, 42(2): 316-330.
7. Weaver, J. and J. Hendler, 2009. "Parallel materialization of the finite RDFS closure for hundreds of millions of triples," in Proc. ISWC, Chantilly, VA, USA, pp: 682-697.
8. Urbani, J., S. Kotoulas, E. Oren and F. Harmelen, 2009. "Scalable distributed reasoning using mapreduce," in Proc. 8th Int. Semantic Web Conf., Chantilly, VA, USA, Oct. 2009, pp: 634-649.
9. Urbani, J., S. Kotoulas, J. Maassen, F.V. Harmelen and H. Bal, 2012. "WebPIE: A web-scale parallel inference engine using mapreduce," J. Web Semantics, 10: 59-75.
10. Billion Triples Challenge 2012 Dataset [Online]. Available:http://km.aifb.kit.edu/projects/btc-2012/
11. RDF Semantics [Online]. Available: http://www.w3.org/TR/rdf-mt/
12. Bo Liu, Kemen Huang, Jianqiang Li and Meng Chu Zhou, 2015. "An Incremental and Distributed Inference Method for Large-Scale Ontologies Based on MapReduce Paradigm", IEEE Transactions on Cybernetics, 45(1): 53-54.
13. Hadoop [Online]. Available: http://hadoop.apache.org/.