International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE)
ISSN: 0976-1353 Volume 21 Issue 1 – APRIL 2016.

SEMANTIC BASED APPROACH TO SCALABLE DISTRIBUTION METHOD FOR LARGE SCALE RDF

Sharmili.G[1],Dr.A.Jebaraj RatnaKumar[2]

[1] *PG Student, Department of Computer Science and Engineering, Karpaga Vinayaga College of Engineering and Technology, Anna University, Chennai – 603 308, Tamil Nadu, India*

[2] *Head of the Department, Department of Computer Science and Engineering, Karpaga Vinayaga College of Engineering and Technology, Anna University, Chennai – 603 308,Tamil Nadu, India*

sharmili.gopi@gmail.com

***Abstract*- The development of Ontologies involves continuous but relatively small modification Semantic information has been reducing but using incremental framework with simple machine learning algorithm. Each level of mapping and reducing based on k-means clustering technique Clustered values have get modification like addition at the end user query has been retrieve with the help of grouped items. Traditional centralized reasoning methods are not sufficient to process large Ontologies. Distributed reasoning methods are thus required to improve the scalability and performance of inferences. This paper proposes an incremental and distributed inference method for large-scale Ontologies by using MapReduce, which realizes high-performance reasoning and runtime searching, especially for incremental knowledge base. By constructing transfer inference forest and effective assertional triples, the storage is largely reduced and the reasoning process is simplified and accelerated.**

***Key Terms*—Big data, MapReduce, ontology reasoning, RDF, Semantic Web**

I.INTRODUCTION

Big data is a term for massive data sets having large, more varied and complex structure with the difficulties of storing, analyzing for further processes or results. The process of research into massive amounts of data to reveal hidden patterns and secret correlations named as big data analytics. The basic idea is that, instead of making a server large, data should be distributed among multiple commodity hardware severs. Map Reduce allows for distributed processing of the map and reduction operations. Provided that each mapping operation is independent of the others, all maps can be performed in parallel through in practice this is limited by the number of independent data sources and o the number of CPUs near each source

Application developers specify the computation in terms of a map and a reduce function, and the underlying MapReduce job scheduling system automatically parallelizes the computation across a cluster of machines.

Hadoop is an open-source implementation of the Google MapReduce programming model. Hadoop consists of the Hadoop common, which provides access to the file systems supported by Hadoop. Particularly, the Hadoop Distributed File Systems (HDFS) provides distributed file storage and is optimized for large immutable blobs of data. A small Hadoop cluster will include a single master and multiple workers nodes. The master node runs multiple processes, including a JobTracker and a NameNode. The JobTracker is responsible for managing running Hadoop cluster. The NameNode, on the other hand, manages the HDFS. The JobTracker and the NameNode are usually collocated on the same physical machine. Other servers in the cluster run a TaskTracker and a DataNode processes. A MapReduce job is divided into tasks. Tasks are managed by the TaskTracker and the DataNode are collocated on the same servers to provide data locality in computation.

MapReduce provides a standardized framework for implementing large-scale distributed computation, namely the big-data applications. There are potential

duplicate computations being performed in this process. However, MapReduce does not have the mechanism to identify such duplicate computations and accelerate job execution. Motivated by this observation, we propose a data-aware cache system for big-data applications using the MapReduce framework. This system aims at extending the MapReduce framework and provisioning a cache layer for efficiently identifying and accessing cache items in a MapReduce job

II.BACKGROUND

Since the data objects in a variety of languages are typically trees, tree pattern matching (twig) is the central issue. Naturally queries in the XML query language specify the patterns of selected predicates on multiple elements which have a tree structured relationships. The complex query tree pattern is usually decomposed into set of basic parent-child and ancestor-descendant relationships. But finding all these basic structural relationships occurrences is a complex process in the XML query processing. There are various techniques provides wireless XML dissemination schemes but none of them supports twig pattern queries since they does not have parent child relationship. Normal index methods divides a query into several sub-queries, thereby join the results together to provide the final answer. Twig pattern search uses tree structures as the master unit of query to avoid expensive join operations and the requirement is used in preprocessing process to include in the child relationship. Intelligent transfer process is to manipulate the concept of the processing to evaluate the computation

A. Hadoop Architecture

Bigdata is often describes as extremely large data sets that have grown beyond the ability to manage and analysis them with traditional data processing tools. The challenges include capture, storage, search, sharing, transfer, analysis and visualization

Bigdata represents the large and rapidly growing volume of information that is mostly untapped by existing analytical applications and data warehousing systems. Organizations are interested in capturing and analyzing this data because it can add significant value to the decision making process. When big-data brings to the business it examines different types of Bigdata and offers suggestions on how to optimize systems infrastructure. It is important to realize that Bigdata comes in many shapes and sizes. It also has many different uses- real time fraud detection, web display advertising and competitive analysis, all centre optimization, social media and sentiment analysis, intelligent traffic management and smart power grids, to name just a few. All of these analytical solutions involve significant (and growing) a volumes of both multi-structure and structured data. Many of these analytical solutions were not possible previously because they were too costly to implement, or because analytical processing technologies were not capable of handling the large volumes of data involved in a timely manner. In some cases, the required data simply did not exist in an electronic form. Deriving inferences in the large-scale RDF files, referred to as large-scale reasoning, poses challenges in three aspects: 1) distributed data on the web make it difficult to acquire appropriate triples for appropriate inferences; 2) the growing amount of information requires scalable computation capabilities for large datasets; and 3) fast processing for inferences is required to satisfy the requirements of online query. Due to the performance limitation of a centralized architecture

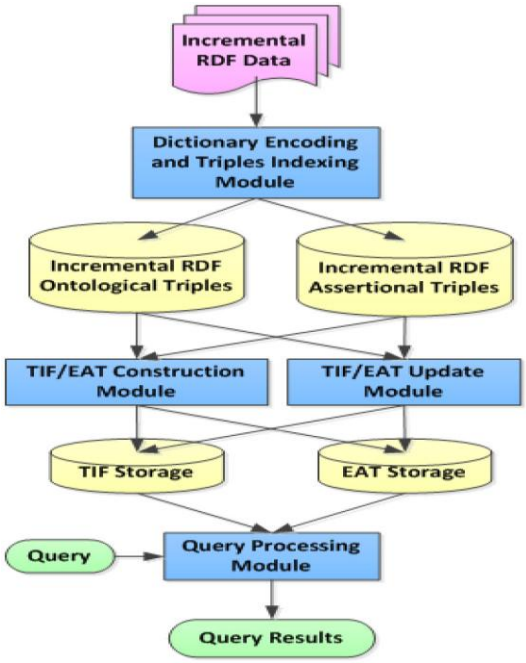

Fig 1 Hadoop Architecture

B. Characteristic

VOLUME: Many factors contribute to the increase in data volume. Transaction based data stored through the years. Unstructured data streaming in form social media. Increasing amounts of sensor and machine-to-machine

data being collected. In the past, excessive data volume was a storage issue. But with decreasing storage costs, other issues emerge, including how to determine relevance within large data volumes and how to use analytics to create value from relevant data

VELOCITY: Data is streaming in at unprecedented speed and must be dealt with in a timely manner. RFID tag, sensors and smart metering are driving the need to deals with torrents of data in near-real time. Reacting quickly enough to deal with data velocity is a challenge for most organizations

VARIETY: Data today comes in all types of formats. Information created from line-of-business applications. Managing, merging and governing different varieties of data are something many organizations

VARIABILITY: In addition to the increasing velocities and varieties of data, data flows can be highly inconsistent with periodic peals. Daily, seasonal and event-triggered peal data loads can be challenging to manage. Even more so with unstructured data involved

COMPLEXITY: Today's data comes from multiple sources and it is still an undertaking to link, match, cleanse and transform data across systems. However, it is necessary to connect and correlate relationships hierarchies and multiple data linkages or your data can quickly spiral out of control

C. TIF/EAT Computation.

- We propose a novel representation method TIF/EAT to support incremental inference over large-scale RDF datasets which can efficiently reduce the storage requirement and simplify the reasoning process
- An efficient and scalable reasoning method called IDIM is presented based on TIF/EAT, and the corresponding searching strategy is given to satisfy end-users' online query needs
- We have implemented a prototype by using the Hadoop platform. It allows one to perform experiments of different methods on billion triples challenge (BTC) benchmark data. A real-world application on healthcare domain is also presented to validate the effectiveness of our method

III. RELATED WORK

Semantic inference has attracted much attention from both Academic and industry nowadays. Many inference engines have been developed to support the reasoning over Semantic For example, Anagnostopoulos and 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 introduced a novel Rule XPM approach that consisted of a concept separation strategy and a semantic inference engine on a multiphase forward-chaining algorithm to solve the semantic inference problem in heterogeneous e-marketplace activities. The SD Type method based on statistical distribution of types in RDF datasets to deal with noisy data presented a temporal extension of the web ontology language (OWL) for expressing time-dependent information. To deal with such large base, some researchers turn to distributed reasoning methods. Weaver and Hendler presented a method for materializing the complete finite RDF closure in a scalable manner and evaluated it on hundreds of millions of triples. Urbani *et al.* proposed a scalable distributed reasoning method for computing the closure of an RDF graph based on MapReduce and implemented it on top of Hadoop. MapReduce-based reasoning and then introduced Map resolve method for more expressive logics. However, these methods considered no influence of increasing data volume, and did not answer how to process users' queries. The storage of RDF closure is thus 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 computation of the entire RDF closure every time when new data arrive. It provide the basic element of Ontologies. To avoid such time-consuming process, incremental reasoning methods are proposed a scalable parallel inference method, named WebPIE, to calculate the RDF closure based on MapReduce for a large-scale RDF dataset. They also adapted their algorithms to process the statements according to their status (existing ones or newly added ones) as incremental reasoning, but the performance of incremental updates was highly dependent on input data. Furthermore, the relationship between newly-arrived data and existing data is not considered and the detailed implementation method is not given presented an incremental reasoning approach based on modules that can reuse the information obtained from the previous versions of Ontology. To speed up the updating process with newly-arrived data and fulfill the requirements of end-users for online queries, this paper presents a method IDIM based on MapReduce and Hadoop, which can well leverage the old and new data to minimize the updating

time and reduce the reasoning time when facing big RDF datasets

IV.PRELIMINARIES

A. Resource Description Framework (RDF)

Semantic Web is based on RDF, which integrates a variety of applications by using extensible markup language (XML) for syntax and universal resource identifier (URI) for naming.RDF is an assertional language intended to be used to express propositions via precise formal vocabularies. An RDF data model is similar to classic conceptual modeling approaches, as it is based on the idea of making statements about resources. The fundamental unit of RDF is a triple that is used to describe the relationship between two things. Its formal definition is <subject, predicate, object>, in which subject denotes a resource, and predicate denotes properties or aspects of the resource and expresses a relationship between the resource and the object. RDF schema (abbreviated as RDFS) is a set of classes with certain properties in RDF. It provides basic elements for the description of Ontologies, or called RDF vocabularies, intended to structure RDF resources the simple protocol and RDF query language RDF closure is a way to realize an RDF query. If the statements in the input Ontology satisfy the conditions in its middle column, a new statement in its right column is added to the ontology. Since the computation of RDF closure is an iterative process, its generation efficiency is notoriously low. In order to distinguish the triples that may trigger the inference on RDFS rules, we divide them into ontological and assertional ones used throughout this paper

Definition 1: Ontological triples are the ones from which significant inferences can be derived, i.e., the triples with predicate rdfs:domain , rdfs:range , rdfs:subClassOf , rdfs:subPropertyOf , and those with predicate rdf:type and object rdfs:Datatype or rdfs:Class or rdfs:ContainerMembershipProperty

Definition 2: Assertional triples are the ones that are not Ontological triples

Table I Rule Statement

Rule Name	If the statement contains:	Then add:
rdfs1	s p o (if o is a literal)	_:n rdf:type rdfs:Literal
rdfs2	p rdfs:domain x & s p o	s rdf:type x
rdfs3	p rdfs:range x & s p o	o rdf:type x
rdfs4a	s p o	s rdf:type rdfs:Resource
rdfs4b	s p o	o rdf:type rdfs:Resource
rdfs5	p rdfs:subPropertyOf q & q rdfs:subPropertyOf r	p rdfs:subPropertyOf r
rdfs6	p rdf:type rdf:Property	p rdfs:subPropertyOf p
rdfs7	s p o & p rdfs:subPropertyOf q	s q o
rdfs8	s rdf:type rdfs:class	s rdfs:subClassOf rdfs:Resource
rdfs9	s rdf:type x & x rdfs:subClassOf y	s rdf:type y
rdfs10	s rdf:type rdfs:Class	s rdfs:subClassOf s
rdfs11	x rdfs:subClassOf y & y rdfs:subClassOf z	x rdfs:subClassOf z
rdfs12	p rdf:type rdfs:ContainerMembershipProperty	p rdfs:subPropertyOf rdfs:member
rdfs13	o rdf:type rdfs:Datatype	o rdfs:subClassOf rdfs:Literal

B. MapReduce

In this paper, our inference method is based on MapReduce and Hadoop platform. MapReduce is a programming model for parallel and distributed processing of batch jobs. Each job contains a map and a reduce, in which the map phase assigns a key to each element and then partitions the input data, while the reduce phase processes each partition in parallel and merges all intermediate values with the same key into final results. 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

V. INCREMENTAL AND DISTRIBUTED INFERENCE OVER LARGE-SCALE RDF DATASET

This section presents the IDIM over large-scale RDF datasets. Before its detailed explanation, gives an overview of its modules and main steps. The input of the system is incremental RDF data files. As our knowledge increases, new RDF data continuously arrive as commonly seen in practice. 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 assertional ones. At the first time that we run the system, the TIF/EAT Construction Module generates the TIF based on the ontological and assertional triples. For the second time and thereafter, the TIF/EAT Update Module only updates

relative TIF and EAT. 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

A. Dictionary Encoding and Triples Indexing

Since RDF data usually contain many statements made of terms that are either URIs or literals, i.e., long sequences of characters, their processing and storage have low performance. Therefore, we use an effective compression method to reduce the data size and increase the application performance. 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 which is implemented in RDF dataset that contain in incremental mapping

B. Reasoning over TIF

In this section, a reasoning method based on TIF is introduced. The forward and reverse paths are first defined in the TIF concept to updated in the map phase design in organization

Definition 3: Forward Path of Edge/Node: In each forest, the forward path of node *n* or edge *r* is a route starting from

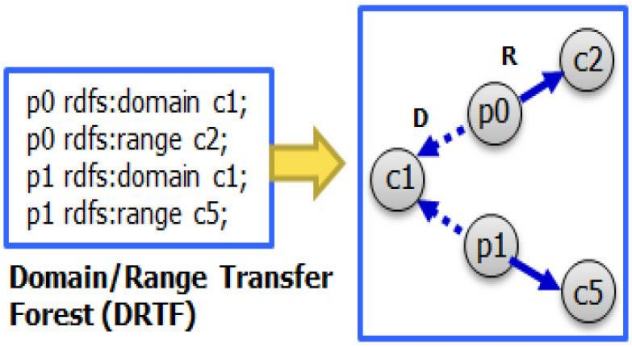

Fig. 2 PTIF Construction

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 derived triple
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

VI. SYSTEM IMPLEMENTATION AND COMPARISON

A. System Architecture

To validate our proposed approaches, a prototype is implemented on the Hadoop platform that is widely used to enable the MapReduce technology. The reasoning by a set of MapReduce programs, 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, PTIF, CTIF, DRTF, PEAT, and CEAT. The Hadoop framework is an open source 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

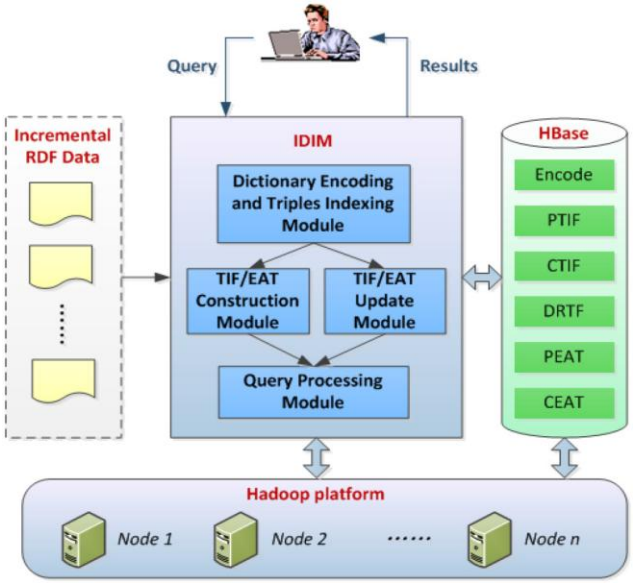

Fig. 3 System Architecture

Table II- Data Sets used in the experiment

TABLE II
BASIC INFORMATION OF BTC DATASET

Dataset	No. of triples	Schema type		
		Domain & Range	Sub-Property	Sub-Class
Datahub	910078982	36338	15068	26146
DBpedia	198090024	1136	0	275
Freebase	101241556	1	0	0
Rest	22328242	2905	746	30373
Timbl	204806751	55086	24431	291095
Overall	1436545555	95466	40245	347889

B. Performance Evaluation

The dataset for our experiment is from the BTC dataset was built to be a realistic representation of the Semantic Web and therefore can be used to infer statistics that are valid for the entire Web of data. BTC consists of five large datasets, Datahub, DBpedia, Freebase, Rest, and Timbl, and each dataset contains several smaller ones. Their overview is shown in Table II. In order to show the performance of our method, we compare IDIM with WebPIE, 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. We run three times of the two methods on each dataset and then calculate the number of the output triples and the time needed for the reasoning. For IDIM, the output triples are the ones in TIF and EAT, and the time for generating TIF/EAT is recorded. For WebPIE, the output triples are the ones in RDF closure, and the time for computing RDF closure is recorded. The result is shown in Table III. From it, we can conclude that the reasoning time for our method is less than WebPIE (76.7% of WebPIE in total time) and the output triples for our method is much fewer than WebPIE (only 61.9% of WebPIE)

.

TABLE III
RESULT FOR THE REASONING (EIGHT NODES)

Dataset	No. of Triples in TIF/EAT	Time for TIF/EAT (min)	No. of Triples in RDF closure	Time for RDF closure (min)
Datahub	713574291	57	1079343655	77
DBpedia	133242743	27	198091689	35
Freebase	94134030	13	101241556	14
Rest	17073633	10	26287842	12
Timbl	114130464	28	326688386	38
Overall	1072155161	135	1731653128	176

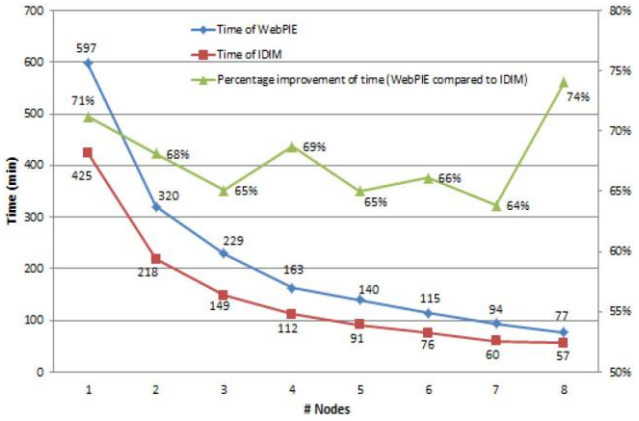

Fig. 4 Processing time on different nodes (Datahub dataset).

VII.CONCLUSION

In the big data era, reasoning on a Web scale becomes increasing 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. The Hadoop configurations are identical to that in IDIM. This paper for the first time proposes an IDIM to deal with large-scale incremental RDF datasets to our best knowledge. The result is scalable and the output triples are the onces in TIF and EAT. The construction of TIF and EAT significantly reduces the computation 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. We have evaluated our system on the BTC benchmark and the results show that our method outperforms related ones in nearly all aspects

REFERENCE

[1] G.Antoniou and A. Bikakis "DR-Prolog: A system for defensible reasoning with rules and Ontologies on the Semantic Web" IEEE Trans Know Data Eng., vol. 19, no. 2, pp. 233–245, Feb. 2007.

[2] J. Cheng, C. Liu, M. C. Zhou, Q. Zeng, and A. Ylä- "Automatic Composition of Semantic Web services based on fuzzy predicate Petrinets," IEEE Trans. Autom Science Eng., Nov. 2013, to be published.

[3] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," Commun. ACM, 2008.

[4] IEEE TRANSACTIONS ON CYBERNETICS,VOL 45, NO . 1, JANUARY 2015

[5] J. Guo, L. Xu, Z. Gong, C.-P. Che, and S. S. Chaudhry,"Semantic Inference on heterogeneous e-Marketplace activities," IEEE Trans. Syst., Man, Cybern. A, Syst,Humans, vol. 42, no. 2, pp. 316– Mar. 2012.

[6] M. J. Iba nez, J. Fabra, P. Álvarez, and J. Ezpeleta, "Model checking analysis of semantically annotated business processes," IEEE Trans. Syst.,Man, Cybern. A, Syst., Humans, vol. 42, no. 4, pp. 854–867, Jul. 2012.

[7] D. Kourtesis, J. M. Alvarez-Rodriguez, and I. Paraskakis, "Semantic based QoS management in systems: Current status and future challenges,"Future Gener. Comput. Syst., vol. 32, pp. 307–323, Mar.2014.

[8] M.S. Marshall, "Emerging practices for mapping and linking life science data using RDF—A case series," Jul. 2012.

[9] M. Nagy and M. Vargas-Vera, "Multiagent Ontology mapping framework for the Semantic Web," IEEE Trans. Syst., Man, Cybern. A, Syst., Humans, vol. 41, no. 4, pp. 693–704, Jul. 2011.

[10] H. Paulheim and C. Bizer, "Type inference on noisy RDF data," in Proc.ISWC, Sydney, NSW, Australia, 2013, pp. 510–525.

[11] V. R. L. Shen, "Correctness in hierarchical knowledge-based requirements,"IEEE Trans. Syst., Man, Cyber. B, Cybern., vol. 30, no. 4,pp. 625–631, Aug. 2000

[12] J. Urbani, S. Kotoulas, E. Oren, and F. Harmelen "Scalable distributed reasoning using MapReduce ," in Proc. 8th Int. Semantic Web Conf.,Chantilly, VA, USA, Oct. 2009, pp. 634–649

[13] J. Urbani, S. Kotoulas, J. Maassen , F. V. Harmelen, and H. Bal,"WebPIE : A web-scale parallel inference engine using"

[14] J. Web Semantics, vol. 10, pp. 59–75, Jan. 2012.

[15] J. Weaver and J. Hendler, "Parallel materialization of the finite RDF Closure for hundreds of millions of triples," in Proc. ISWC, Chantilly,VA , USA, 2009, pp. 682–697