International Journal of Grid and High Performance Computing
Volume 8 • Issue 3 • July-September 2016

Efficient Querying Distributed Big-XML Data using MapReduce

Song Kunfang, Huazhong University of Science and Technology, Wuhan, China
Hongwei Lu, Huazhong University of Science and Technology, Wuhan, China

ABSTRACT

MapReduce is a widely adopted computing framework for data-intensive applications running on clusters. This paper proposed an approach to exploit data parallelisms in XML processing using MapReduce in Hadoop. The authors' solution seamlessly integrates data storage, labeling, indexing, and parallel queries to process a massive amount of XML data. Specifically, the authors introduce an SDN labeling algorithm and a distributed hierarchical index using DHTs. More importantly, an advanced two-phase MapReduce solution are designed that is able to efficiently address the issues of labeling, indexing, and query processing on big XML data. The experimental results show the efficiency and effectiveness of the proposed parallel XML data approach using Hadoop.

KEYWORDS

B-SLCA, Big XML, Distributed Programming, MapReduce, Parallel Programming

INTRODUCTION

XML processing has been extensively studied in the literature. The XML operator typically includes labeling, indexing, and keywords searching, among which labeling and indexing are two important components. Since semantics are defined using the notion of lowest common ancestor (LCA), at the heart of existing query algorithms is the Dewey labeling (Xu, Ling, Wu & Bao, 2009). The Dewey label of a node u is a concatenation of all its ancestor nodes' local label on the path from the document root to v. Much attention has been paid to keywords searching on XML files. It is demanding to design efficient query processing methods for keyword searching on XML data, because XML applications require fast query performance to meet the needs of a large number of users. To improve XML processing speed in the MapReduce framework, we design a *sequence depth number* or SDN labeling, a flexible indexing model using the distributed hash table or DHT.

This study is focused on XML files that adopt the standard XML format, where each file is characterized as an ordered, rooted, and labeled tree (Quan & Moon, 2001). Each edge represents an element-element relationship or an element-value relationship. Each element is identified by a pair of start-tag and end-tag; elements may have attributes with their values. If keyword k appears at least once in one of a node name, an attribute name, and text value of root node v, we say v directly contains k.

To speed up the query process, each node is usually assigned with a label uniquely representing v; the label can be used to compute positional relationships. Most existing labeling methods are assigned with the Dewey encoding. In our solution, we assign each node with a sequence depth number (SDN) that is compatible with the XML document order using a parallel processing technique. All labeled nodes are stored in DHTs on the Hadoop distributed file system or HDFS; the tag name is the key and the text value with prefix label is the value.

DOI: 10.4018/IJGHPC.2016070105

More concretely, the contributions of this paper can be summarized as follows:

- We develop the SDN labeling technique for each element in *Hadoop distributed file system* and construct a flexible indexing model based on DHTs, thereby improving query performance of XML datasets stored in HDFS;
- We design an efficient query process in the form of two MapReduce jobs, and the B-SLCA keyword search approach with SDN label in DHTs is developed, which is a bottom-up retrieval way to quickly find an SLCA node.

Related Work

During the past few years, a handful of efficient approaches have been proposed for XML processing evaluation, which includes three key operation parts, labeling (Choi, Lee, & Lee, 2014; Zhou, Bao, & Meng, 2013; Xu, Ling, Wu, & Bao, 2009), indexing (Camacho-Rodriguez, Colazzo, & Manolescu, 2012; Chen, Vo, & Ooi, 2011; Hsu, Liao, & Shih, 2012; Ottaviano, & Grossi, 2011), and keyword searching (Feng, & Li, 2012; Li, Li, & Zhou, 2009). Much attention has been paid to keyword searching on XML data. Efficient query processing of large XML data plays an important role in keyword searching, because various applications depend on fast query performance to simultaneously support an enormous number of users. The most widely adopted semantics are arguably the smallest lowest common ancestor or SLCA (Chen, & Papakonstantinou, 2010; Ling, & Xu, 2012; Hsu, Liao, & Shih, 2012) and exclusive lowest common ancestor or ELCA.

Choi et al. proposed an efficient method to parallelize conventional tree labeling algorithms with the MapReduce programming model (Choi, Lee, & Lee, 2014). All elements are labeled with the two prominent tree labeling schemes, namely, the interval-based labeling scheme and the prefix-based labeling scheme, which naturally group elements by their tag names. Choi et al. also implemented data repartition techniques balancing workload of the runtime system. Xu et al. developed a novel dynamic Dewey labeling scheme called DDE (Xu, Ling, Wu, & Bao, 2009), which is tailored to process both static and dynamic XML document; Compact DDE (CDDE) was introduce to optimize the performance of DDE for insertions. These two labeling methods adopt the Dewey labeling scheme to facilitate efficient updates and query processing.

Chen et al. developed the DaaS framework supporting DBMS-like indexes in the cloud (Chen, Vo, & Ooi, 2011). They proposed self-tuning strategies to optimize the indexing performance over the generic clayey overlay, thereby achieving high performance and scalability. Zhou et al. assigned each node a unique ID, which is compatible with a document order (Zhou, Bao, & Meng, 2013). They proposed a new type of inverted index named IDList. To solve a variant of the set intersection problem, Zhou et al. provided SLCA/ELCA computation and applied additional hash indices on IDLists. They designed several algorithms to accelerate the SLCA/ELCA computation, thereby improving the overall performance.

SDN LABELING AND INVERTED INDEXING

In processing of XML data, a challenge lies in the fact that a start-tag and its corresponding end-tag may not be placed together in the same chunk. To address this problem, we exploit an additional split information table called *SIT* to build complete indices for partitioned tags in blocks, which record mismatching tags and their levels.

Related Content

Pricing Computational Resources in Grid Economies

Kurt Vanmechelen, Jan Broeckhove, Wim Depoorter and Khalid Abdelkader (2009). *Handbook of Research on Grid Technologies and Utility Computing: Concepts for Managing Large-Scale Applications (pp. 170-182).*

www.igi-global.com/chapter/pricing-computational-resources-grid-economies/20519?camid=4v1a

Balanced Job Scheduling Based on Ant Algorithm for Grid Network

Nikolaos Preve (2012). *Evolving Developments in Grid and Cloud Computing: Advancing Research (pp. 13-30).*

www.igi-global.com/chapter/balanced-job-scheduling-based-ant/61980?camid=4v1a