International Journal of Database Theory and Application
Vol.6, No.5 (2013), pp.85-94
http://dx.doi.org/10.14257/ijdta.2013.6.5.08

Research on Vector Spatial Data Storage Schema Based on Hadoop Platform

Kun Zheng and Yanli Fu

Faculty of Information Engineering,
China University of Geoscience, Wuhan 430074, China
michael_power@sina.com, fuyli6@126.com

Abstract

Cloud computing technology is changing the mode of the spatial information industry which is applied and provides new ideas for it. Since Hadoop platform provides easy expansion, high performance, high fault tolerance and other advantages, we propose a novel vector spatial data storage schema based on it to solve the problems on how to use cloud computing technology to directly manage spatial data and present data topological relations. Firstly, vector spatial data storage schema is designed based on column-oriented storage structures and key/value mapping to express spatial topological relations. Secondly, we design middleware and merge with vector spatial data storage schema in order to directly store spatial data and present geospatial data access refinement schemes based on GeoTools toolkit. Thirdly, we verify the middleware and the data storage schema through Hadoop cluster experiments. Comprehensive experiments demonstrate that our proposal is efficient and applicable to directly storing large-scale vector spatial data and timely express spatial topological relations.

Keywords: *Vector Spatial Data StorageSchema, HDFS, HBase, GeoTools*

1. Introduction

Spatial data is the basis of GIS applications. The technological change in the spatial data management would make great advancement in GIS. With the advancements of data acquisition techniques, large amounts of geospatial data have been collected from multiple data sources, such as satellite observations, remotely sensed imagery, aerial photography, and model simulations. The geospatial data are growing exponentially to PB (Petabyte) scale even EB (Exabyte) scale [1, 2]. As this presents a great challenge to the traditional database storage, especially in terms of vector spatial data storage due to its complex structure, the traditional spatial database storage is facing a series of questions such as poor scalability and low efficiency of data storage.

With the superiority in scalability and data storage efficiency, Hadoop, and large-scale distributed data management platform in general, provides an efficient way for large-scale vector spatial data storage. Many scholars have done some research on data storage based on Cloud computing technology. Xuhui liu, *etc* [3, 4] researched large number of small files low efficiency problem based on Hadoop platform and tried to integrate many small files into a large file in order to improve reading and writing performance of a large number of small files of spatial data. Ariel Cary, *etc* [5] applied the MapReduce model to process spatial data. Yonggang Wang, *etc* [6] researched geospatial data storage, geospatial data index based on Hadoop platform. And they compared Hadoop platform with Oracle Spatial database in attribute data query and concluded that Hadoop is more efficient in data query. Jifeng Cui, *etc*

ISSN: 2005-4270 IJDTA

[7] studied the heterogeneous geospatial data organization storage based on Google's GFS to solve the problem of multi-source geospatial data storage and query efficiency. But there is rarely relevant research on how to use unstructured database to directly store spatial data. In this paper, we studied the storage model of vector spatial data based on Hadoop platform and provided an efficient way for large-scale vector spatial data storage to lay the theoretical foundation.

2. Data Storage based on Hadoop

Hadoop, and large-scale distributed data processing in general, is rapidly becoming an important skill sets for many programmers [8]. It is the core composition of distributed file system HDFS, distributed unstructured database HBase and distributed parallel computing framework MapReduce. The key to the distinctions of Hadoop is accessible, robust and scalable. Today, Hadoop is a core part of the computing infrastructure for many web companies, such as Yahoo, Facebook, LinkedIn, and Twitter. Many more traditional businesses, such as media and telecom, are beginning to adopt this system, too [8].

2.1. Distributed File System HDFS

HDFS, and Hadoop Distributed File System, is the main application of the Hadoop distributed file system and is shown in Figure 1 [9]. And it stores data in blocks. Each block is the same size and its default size is 64 MB. A HDFS cluster is the composition of NameNode and a number of DataNodes. Clients firstly read metadata information through NameNode, and then access the data through the appropriate DataNode. If clients store data in the DataNode, NameNode would record the metadata information of data [10].

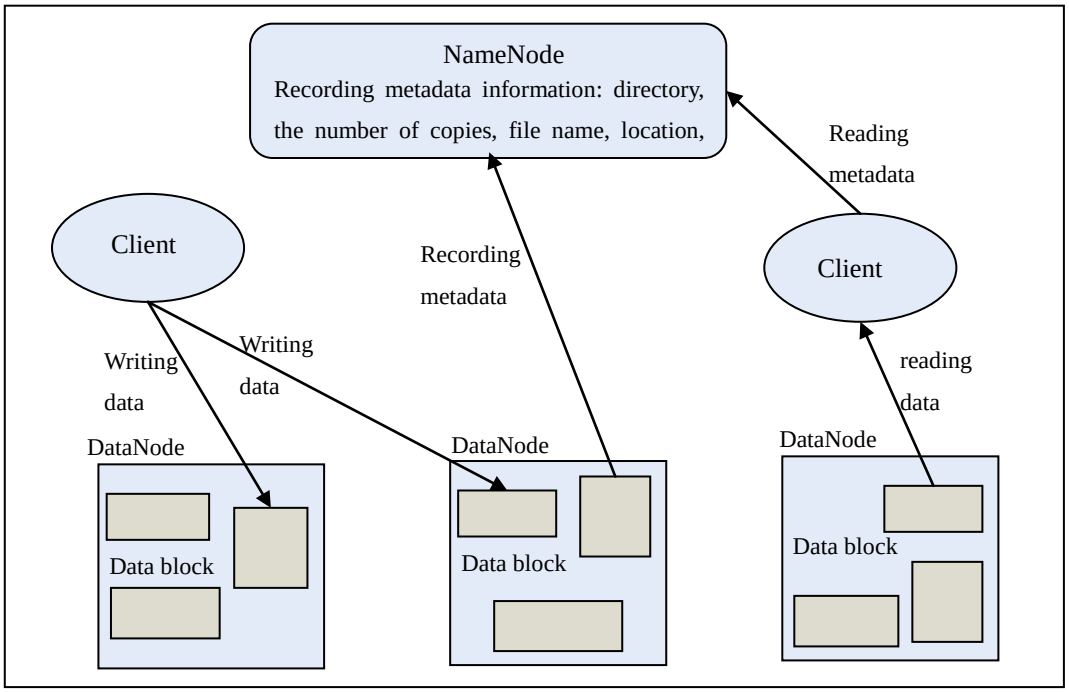

Figure 1. The Architecture of HDFS [9]

2.2. HBase Database's Storage Mechanism

HBase is not a column-oriented database in the typical RDBMS sense, but utilizes an on-disk column storage format and stores data between key-value database and traditional relational database. It can use the local file system and HDFS. But HBase has the ability of processing data and improving data reliability and the robustness of the system if it uses HDFS as its file system. The relationship between HBase and HDFS is shown in Figure 2. In this paper, HBase database stores data based on HDFS.

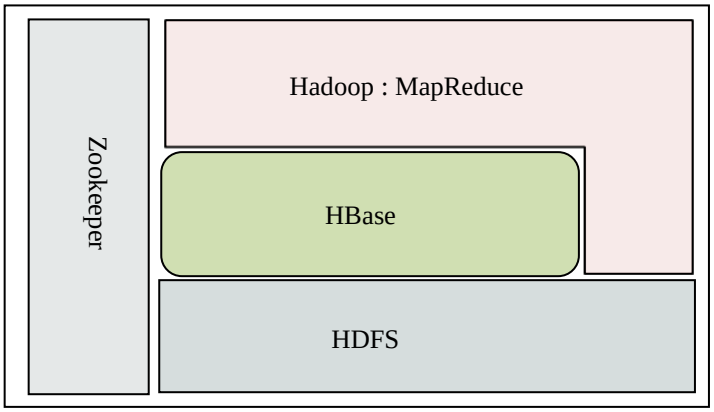

Figure 2. The Relationship between HBase and HDFS

HBase stores data on disk in a column-oriented format, and it is distinctly different from traditional columnar databases: whereas columnar databases excel at providing real-time analytical access to data, HBase excels at providing key-based access to a specific cell of data, or a sequential range of cells [10]. Each row of the same table can have very different column and each column has a time version that is called timestamp. Timestamp records the update of database that indicates an updated version. The logical view of the HBase database has two columns family: c1 and c2 that are shown by the Table 1. Each row of data expresses the update through timestamp. The smaller the numbers of timestamp is, the newer the data is. In the physical storage, the blank column on the logic view of HBase could not be stored actually. Each column cluster is saved through several files and different column clusters are stored separately. The feature that is different from traditionally row-oriented database.

Table 1. HBase Storage Data Logic View

RowKey	Time Stamp	Column Family:c1		Column Family:c2	
		info	value	Attribute	value
r1	t6	c1:1	Value1		
	t5	c1:2	Value2	c2:1	Value1
	t4			c2:2	Value2
	t3				
r2	t2	c1:1	Value1		
	t1	c1:2	Value2		

In a row-oriented system, indexes tables as additional structures are built to get quickly results for queries. But HBase database does not need additional storage for data indexes, because data is stored with the indexes itself. Data loading is executed faster in a column-oriented system than in a row-oriented one. As we all know, all data of each row is stored together in a row-oriented and all data of in a column is stored together in a

column-oriented database. And column-oriented system makes all columns parallel load to reduce the time of data loading.

3. Designing Vector Spatial Data Storage Schema

Vector data is more complex than raster data in organization because it not only considers the scale, layers, point, line, surface and other factors but also involves complex spatial topological relations. We should design vector data storage schema that fits on the Hadoop distributed platform in order to take advantage of Hadoop storage. This schema would offer an efficient organization and complete the storage of vector spatial data in the unstructured database platform.

3.1. Vector Spatial Object Model

OGC(R) standards are technical documents that detail interfaces or encodings. Software developers use these documents to build open interfaces and encodings into their products and services [12]. OGC simple factor model that is proposed by International Association of OpenGIS is shown in Figure 3 to share geospatial information and geospatial services [13, 14]. In this paper, we use OGC simple factor model to design vector object model in order to achieve the better interoperability of heterogeneous spatial database.

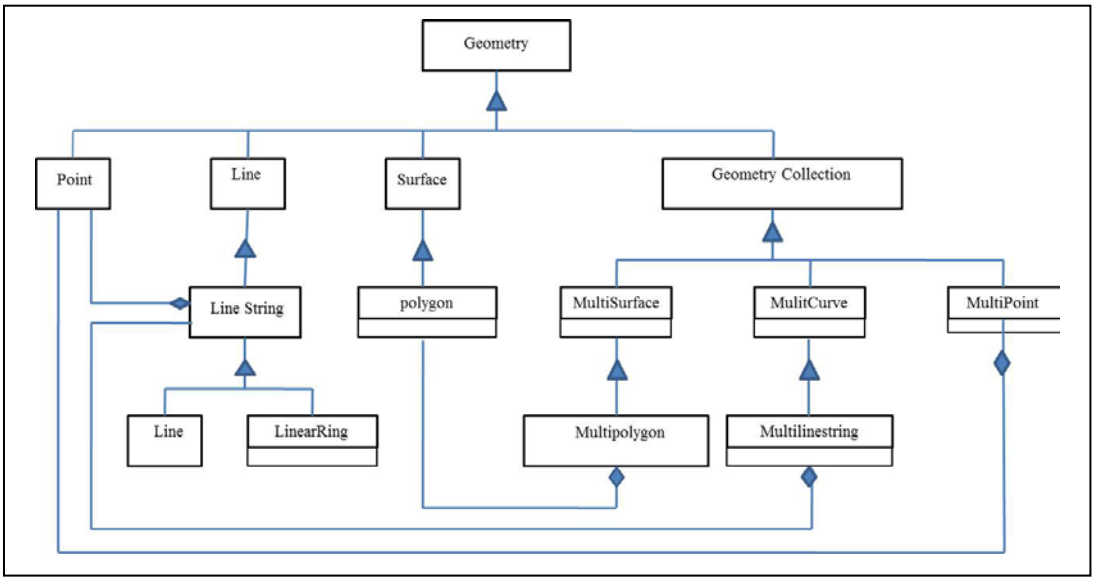

Figure 3. OGC Simple Feature Code Model

3.2. Vector Spatial Data Logical Storage

Vector data consists of coordinate data, attribute data, topology data [15]. We designed vector spatial data storage schema based on the HBase database storage model according to the characteristics of vector data. Vector spatial data logical storage schema is shown by the Table 2 and it contains three columns family which are coordinate, attribute, topology, respectively recording coordinate information, attribute information and topology information of data. Each data type in the storage system is string and is parsed into appropriate data type in accordance with the Table 3 that is the dictionary storage structure of vector data type

when is used. We can cleverly design RowKey in accordance with the actual situation and usage scenarios to obtain a collection of the results in the query and good performance.

The following is important code to establish the table structure of vector spatial data storage. The variable tableName represents the table's name and the variable familys contains some columns family in a table:

```
public static void creatTable(String tableName, String[] familys) throws Exception {
        HBaseAdmin admin = new HBaseAdmin(conf);
      if (admin.tableExists(tableName)) {
          System.out.println("table already exists!");
      } else{
          HTableDescriptor tableDesc = new HTableDescriptor(tableName);
       for (int i=0; i<familys.length; i++ ){
            tableDesc.addFamily(new HColumnDescriptor(familys[i]));          }
       admin.createTable(tableDesc);          }
```

Table 2. Vector Spatial Data Storage View

RowKey	TimeStamp	Column Family: coordinate		Column Family: Attribute		Column Family: topology	
		info	value	Attribute	value	Topology	value
Fea_ID1	T8	Info:1	(x, y)				
	T7	Info:2	coordinate				
	T6			Attribute:1	Value1		
	T5			Attribute:2	Value2		
	T4						
	T3					Topo:1	Value1
Fea_ID2	T2	Info:1	(x, y)				
	T1	Info:2	coordinate				

Table 3. The Dictionary Storage Structure for Vector Data Types

Row Key	Time Stamp	Column Family: Coordinate		Column Family: Attribute		Column Family: Topology	
		info	value	Attribute	value	Topology	value
Fea_ID1	T8	"(x, y)"	"double"				
	T7	"Coordinate"	"string"				
	T6			"Attribute1"	"int"		
	T5			"Attribute2"	"string"		
	T3					"Topo1"	"int"

3.3. Vector Spatial Data Physical Storage

HBase stores data on disk in a column-oriented format although the logical view consists of many lines. The physical storage that the RowKey is Fea_ID1 in the Table 2 is shown by the Table 4, Table 5 and Table 6. From these tables, it concluded that the blank column on the logic view in Table 2 could not be stored on the physical model actually. It is different from relational database when we design data storage model and develop procedure.

And in HBase database, we don't need built additional indexes, so data is stored within indexes themselves. In the data query, Vector spatial data storage model based on HBase database only read the columns required in a query. This queried way makes it provide a better performance for analytical request.

Table 4. The Physical Storage of Coordinate Column

Row Key	Time Stamp	Column Family: Coordinate info	value
Fea_ID1	T8	Info:1	(x, y)
	T7	Info:2	coordinate

Table 5. The Physical Storage of Attribute Column

Row Key	Time Stamp	Column Family: Attribute	Attribute value
Fea_ID1	T6	"Attribute1"	"int"
	T5	"Attribute2"	"string"

Table 6. The Physical Storage of Topology Column

Row Key	Time Stamp	Column Family: Topology Topology	value
Fea_ID1	T3	Topo:1	Value1

4. Research Methodology

4.1. Experimental Environment

We used 2 computers as hosts to clone 3 computers by installing VMware8.0 virtual machine in order to verify the vector spatial data storage schema and develop the middleware. We deploy HDFS, HBase and MapReudce architecture in a virtual machine as the Master and SlaveNode, install Zookeeper in a virtual machine as Zookeeper SlaveNode and deploy RegionServer in every virtual machine as SlaveNode. This configuration of the structure is shown in Figure 4. The operating system of virtual machine is RedHat Enterprise 5, memory 1GB, Hadoop-0.20.2, HBase-0.92.0, jdk1.6.0-38 and development environment contains Eclipse IDE for Java EE Developers. Experimental data is 1:50000 vector data.

In Figure 4, Master is responsible for SlaveNode load balancing. It uses RegionServer to adjust the data block distribution and migrates the data block from invalid RegionServer when SlaveNode shut down. Zookeeper monitors the SlaveNodes' health status and gives feedback about their health status to Master. RegionServer records the metadata information that SlaveNodes store and regularly finishes SlaveNodes' cache data. Client obtains data from SlaveNode by RegionServe and uses Region.flushcache ( ) method to write the contents of

the cache to the HDFS file. SlaveNode is responsible for user's request the management of data. It reads and writes data to the HDFS file system.

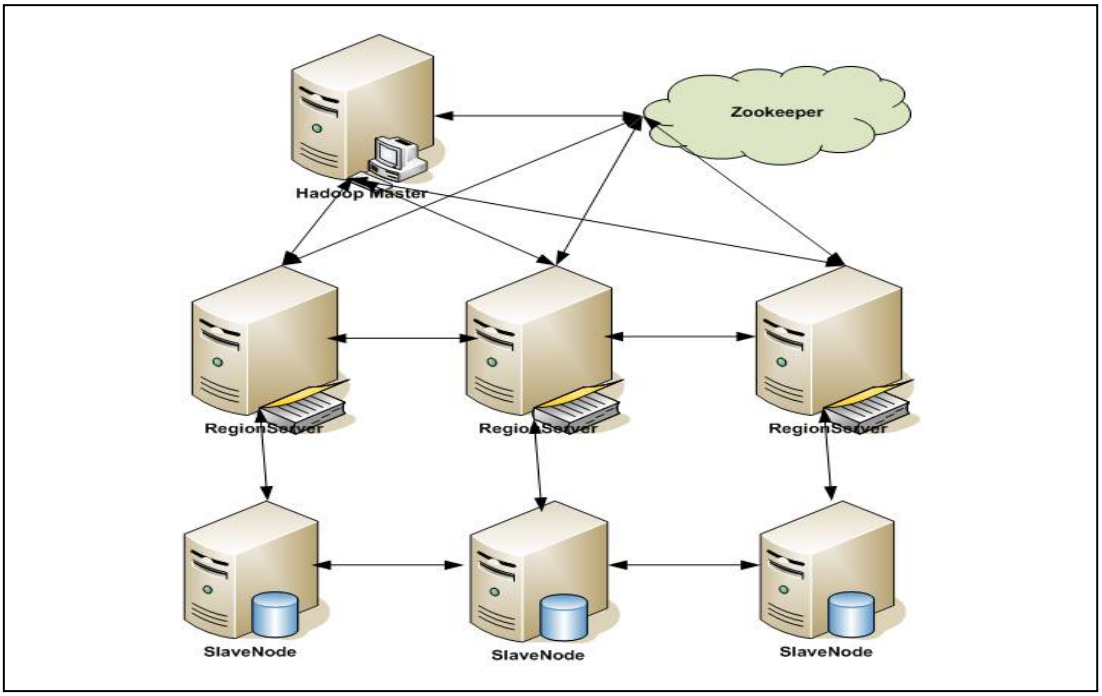

Figure 4. The Architecture in Experimental Environment

4.2. Developing Middleware Based on GeoTools

Due to expensive cost of commercial GIS software, GeoTools is an open source GIS toolkit developed from free soft foundation with the Java language code. And it contains a lot of open source GIS projects and standards-based GIS interface, provides a lot of GIS algorithm, and possesses a good performance in the reading and writing of various data formats [16, 17]. In this experiment, we use GeoTools-2.7.5 open source project to read shapefile data from client and make the appropriate conversion to use the put () method to import data into HBase database by DataStore, FeatureSource, FeatureCollection class libraries. And we design middleware and develop these methods and vector spatial data storage schema to access and display the vector spatial data based on GeoTools toolkit.

According to HBase database query mechanism, we use get ( ) and san ( ) method to search data from database. Get ( ) method acquires a single record and scan ( ) method requires range queries on spatial data by limiting setStartRow ( ) and setStopRow ( ). The important code is as follows.

```
//Get() reading a single record
HTable table = new HTable(conf, tableName);
Get get = new Get(rowKey.getBytes());
Result rs = table.get(get);
Bytes[] ret=rs.getValue((Familys+":"+Column)) ;
//Scan() requiring range queries
```

```
HTable table = new HTable(conf, tableName);
Scan s = new Scan();
s.setStartRow(startRow);
s.setStopRow(stopRow);
ResultScanner    rs = table.getScanner(s);
```

4.3. Experimental Results

In Hadoop client, we use the scan ( ) method to query the J48E023023 road layer data from 1:50000 vector data. To complete the inquiry process Hadoop platform takes 1.26 seconds, while the Oracle Spatial platform takes 1.34 seconds. Due to Hadoop platform is designed to manage large files, it suffers performance penalty while managing a large amount of data. It can be seen that the efficiency of data storage is not too high because the amount of data is too small. But HBase database is used to manage massive spatial data efficiently. And it expands nodes to obtain more storage space and improve computational efficiency.

Finally, we use the middleware to develop the Feature ( ), FeatureBuilder ( ), FeatureCollection ( ), ShapefileDataStore ( ) class libraries to create shapefile for the reading data in order to show this data by the middleware. And the query data is shown in Figure 5.

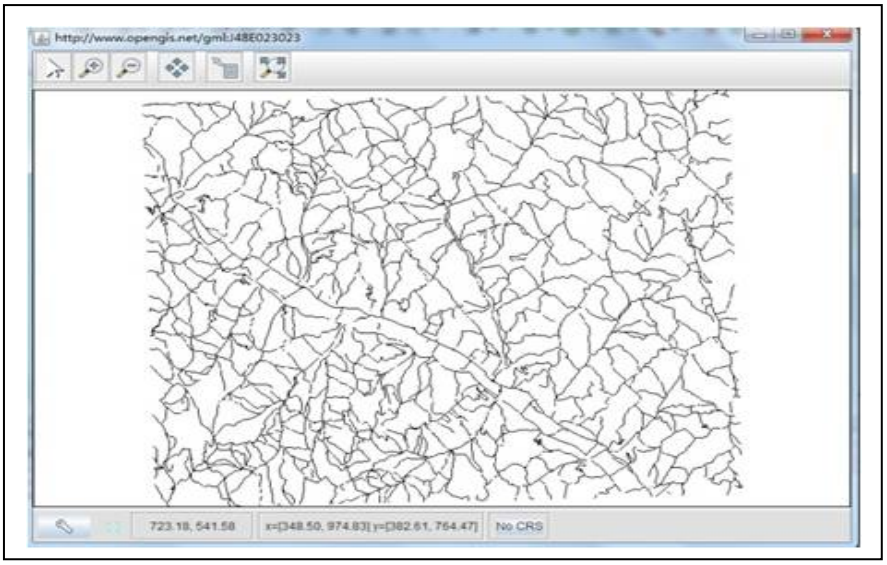

Figure 5. Road Layer Data Shown

Comprehensive experiments demonstrate the effectiveness and availability of the vector spatial data storage schema to solve the problems that Hadoop platform cannot directly store vector spatial data and present data topological relations. In the above experiment, we can use the Hadoop platform to efficiently store and manage vector spatial data. And it provides a new and effective way for large-scale vector spatial data storage.

5. Conclusions and Future Work

In this paper, we analyze HDFS distributed file system and HBase distributed database storage mechanism and offer the vector spatial data storage schema based on Hadoop open

source distributed cloud storage platform. Finally, we design middleware to merge with vector spatial data storage schema and verify the effectiveness and availability of the vector spatial data storage schema through the experiment. And it also provides an effective way for large-scale vector spatial data storage and for many companies which are committed to study Hadoop to store large-scale data. Theoretically, according to Hadoop data storage strategy, we overcome poor scalability, low efficiency and other problems of traditional relational database to provide unlimited storage space and high reading and writing performance for large-scale spatial data. But we should design excellent spatial data partition strategy and build distributed spatial index structure with high performance to efficiently manage large-scale spatial data. Future work should look at spatial data partition strategy and distributed spatial index structure with the goal of further enhancing data management effectiveness.

Acknowledgments

This research is partially supported by the National Technology Research Program of the Ministry of Science and Technology of China (No.2012BAH35B01) and the Support program for Disciplinary Leaders Plan in Wuhan (No.201271130443) .We would also like to thank the anonymous reviewers for their valuable comments.

References

[1] Y. Zhong, J. Han, T. Zhang and J. Fang, "A distributed geospatial data storage and processing framework for large-scale WebGIS", 20th International Conference on Geoinformatics (GEOINFORMATICS), Hong Kong China, **(2012)** June 15-17.
[2] S. Sakr, A. Liu, D. M. Batista and M. Alomari, "A Survey of Large Scale Data Management Approaches in Cloud Environments", Communications Surveys & Tutorials, vol. 13, no. 3, **(2011)**, pp. 311-336.
[3] X. H. Liu, J. Han and Y. Zhong, "Implementing WebGIS on Hadoop: A case study of improving small file I/O performance on HDFS", IEEE International Conference on Cluster Computing and Workshops, New Orleans, Louisiana, **(2009)** August 31- September 4, pp. 1- 8.
[4] D.-W. Zhang, F.-Q. Sun, X. Cheng and C. Liu, "Research on hadoop-based enterprise file cloud storage system", 3rd International Conference on in Awareness Science and Technology (iCAST), Dalian China, **(2011)** September 27-30, pp. 434-437.
[5] A. Cary, Z. Sun, V. Hristidis and N. Rishe, "Experiences on Processing Spatial Data with MapReduce, the 21st International Conference on Scientific and Statistical Database Management", New Orleans, LA, USA, **(2009)** June 02-04, pp. 1-18.
[6] Y. Gang Wang and S. Wang, "Research and Implementation on Spatial Data Storage and Operation Based on Hadoop Platform", Second IITA International Conference on Geoscience and Remote Sensing, Qingdao China, **(2010)** August 28-31, pp. 275-278.
[7] J. Cui, C. Li, C. Xing and Y. Zhang, "The framework of a distributed file system for geospatial data management", Proceedings of IEEE CCIS, **(2011)**, pp. 183-187.
[8] C. Lam, "Hadoop in Action", Manning Publications, **(2010)**.
[9] K. Shvachko, K. Hairong, S. Radia and R. Chansler, "The Hadoop Distributed File System", IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST), Washington, DC: IEEE Computer Society, **(2010)** May 3-7, pp. 1-10
[10] D. Borthakur, "The Hadoop Distributed File System: Architecture and design", **(2008)**.
[11] M. Loukides and J. Steele, "HBase: The Definitive Guide", Published by O'Reilly Media: First Edition, **(2009)** September.
[12] OGC, http://www.opengeospatial.org/standards, **(2012)** April 31.
[13] A. Rao, G. S. Percivall and Y. Enloe, "Overview of the OGC catalog interface specification", IEEE 2000 International on Geoscience and Remote Sensing Symposium, Hawaii USA, **(2000)** July 24-28, pp. 1211-1233.
[14] GeoTools, http://www.geotools.org , **(2012)** January 31.
[15] X. Wu, "Spatial database", Beijing: Science Press, **(2009)**.
[16] J. Tan, J. Xu and L. Wan, "Open source approach for Internet GIS and its application", Second International Symposium on Intelligent Information Technology Application, **(2008)**, pp. 264-268.

[17] X. Li, L. Di, W. Han, Z. Peisheng and U. Dadi, “Sharing and reuse of service-based geospatial processing through a Web Processing Service”, 17th International Conference on in Geoinformatics, Shanghai, China, **(2009)** June 24-16, pp. 1-5.

