Research on Storage and Processing of MongoDB for Laser Point Cloud under Distribution

Xu Xudong [1, a] and Guo Rui[1,b]

[1] College of Computer Science Beijing University of Technology, Beijing 100124, China

[a] xuxudong@bjut.edu.cn , [b] 1045700084@qq.com

Keywords: Distribution, MongoDB, Laser point cloud, Sharding cluster, Range based partitioningHash, based partitioning

Abstract. In recent years, the application of laser point cloud data has increased dramatically. How to efficiently store and fast process the data becomes an important research direction at present. Point cloud data contain a wealth of geographic information, belonging to the category of spatial data. Traditional relational databases are relatively weak in massive spatial data storage and processing, while the application of non-relational databases provides a new perspective of study for this fact. Sharding technology is a solution for database level extension. In this thesis, sharding cluster for MongoDB is established under distributed environment, while distributed storage, spatial query and MapReduce operation test for numerous laser-point cloud data will be implemented through scope sharding and Hash-based sharding, which completely reflects huge advantages of MongoDB under distribution in storage and processing for spatial data.

1 Overview

As information technology develops, construction of digital city has been an overwhelming trend, and GIS (Geographic Information System) has been applied more and more widely in all industries. In recent years, laser-point cloud data rapidly increase, and the point cloud data obtained by airborne, vehicle-mounted and ground laser scanning system have been reached TB, and even PB level [1]. How to realize high-efficient and safe storage and rapid & simple processing has been an important study aspect at present.

The laser-point cloud data includes rich geographic information such as three-dimensional coordinate, color and reflection strength, belonging to domain of spatial data. The advantages of traditional relationship database reflect on data completeness and consistency problem, but have paid less attention to storage and processing for spatial data [2]. However, the non-relationship database can dispose many semi-structural and non-structural data under distributed environment, and show huge advantages on storage and processing for spatial database [3].

As a typical non-relationship database, MongoDB is a document type database with high performance, open source and free mode and supports Bson data structure which is a binary system storage format similar to Json [4]. With huge and strong query language, MongoDB can almost realize all functions of statement query for similar relationship database, and provide complete index support for data; besides, its native support for spatial index shows obvious advantages in disposing geographical location information[5]. In addition, MongoDB also provides many functions such as data redundancy, fault transfer and automatic sharding. So in this thesis, sharding cluster for MongoDB is established under distributed environment, while distributed storage, spatial query and MapReduce operation test for numerous laser-point cloud data will be implemented through scope sharding and Hash-based sharding, which completely reflects huge advantages of MongoDB under distribution in storage and processing for spatial data.

2 Sharding cluster for MongoDB

Sharding can split data level to different physical nodes, and break through I/O capacity limitation for single node server, being a solution for extending database level [6]. MongoDB has strong Auto Sharding function, can dynamically add or delete node, balance loads and provide fault transfer function [7]. Sharding cluster for MongoDB mainly includes three parts, i.e. Shard node, config node and mongos node, each type of nodes can be made up of one or more computers.

Shard node can be used to store actual data. It can be a single mongod example or Replica Set (replica set configuration) [8]. config node is stored with metadata, including cluster configuration information and data location. The main function for mongos node is data routing, of which any data and metadata will not be stored, but users can use client and drive to visit. To ensure data backup, auto fault transfer and recovery capacity for Sharding cluster, each Shard node in this thesis is a Replica Set. See Figure 1 for environmental framework of Replica Set+Sharding:

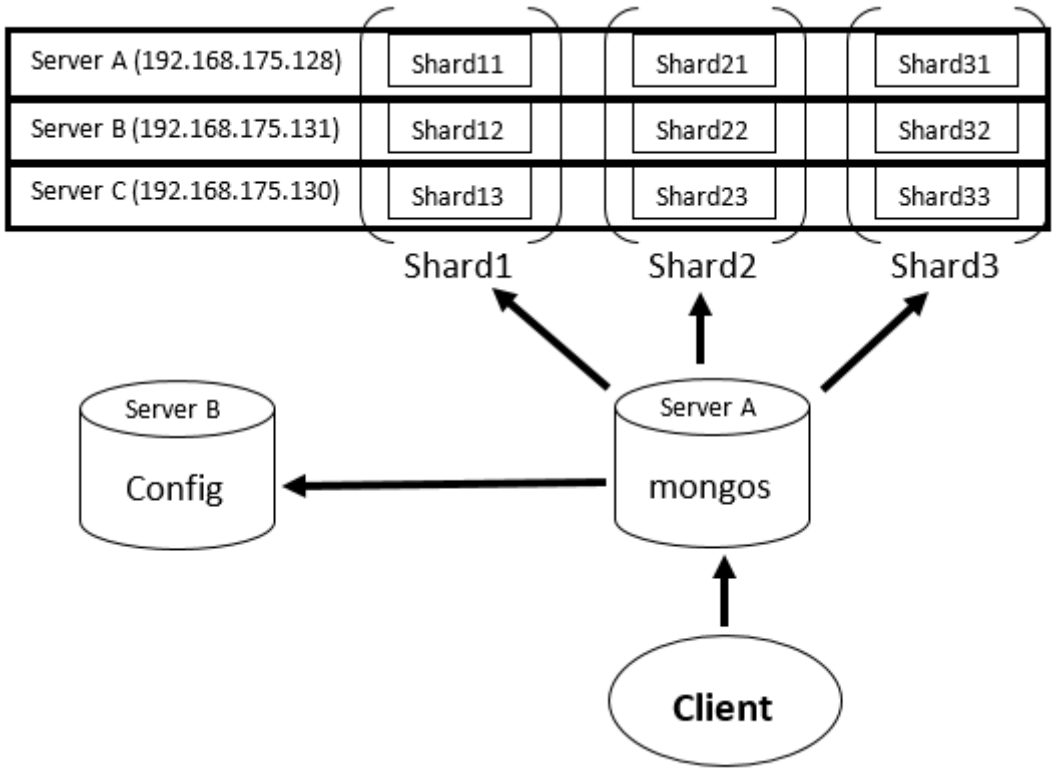

Fig. 1 Environmental Architecture of Replica Set and Sharding

3 Performance test for Sharding cluster

Due to limit of experiment conditions, Vmware Workstation 11 is applied in this thesis to set up Sharding cluster, and three servers (Server A, Server B, Server C) are configured uniformly, with 1G internal storage and 20G HDD, Ubuntu14.04 OS and mongodb-linux-x86_64-2.4.6 database. Shell command and script language JavaScript are used to conduct scope sharding and Hash-based sharding and test storage and processing performance for mass laser-point cloud data.

3.1 Distributed storage for laser-point cloud data

Sharding cluster is to distribute and store data in collection level. To conduct sharding for a set of data, a proper Shard Key shall be selected firstly [9], then scope sharding or Hash-based sharding will be applied to split data into several chunks and distribute in shard node. MongoDB2.4 above version supports Hash-based sharding. The format of laser-point cloud data is as follows:

{

```
    "_id" : ObjectId("56a9dc5676d3f3b90a1cf6f4"),
    "X" : 292730.639,
    "Y" : 180802.526,
    "Z" : 19.655,
    "R" : 36,
    "G" : 74,
    "B" : 145,
    "Loc" : [
        178.78388,
        55.05072
    ]
}
```

_id is automatically generated by MongoDB, and each set is provided with unique _id value to ensure unique identification for each document in the set. X, Y, Z and R, G, B are three-dimensional coordinates and color information for point cloud data. Loc is the array object made up of precision and latitude and refers to geographical location information for point cloud data. In this thesis, the distributed storage is implemented for 1w, 5w, 10w, 50w and 100w laser-point cloud data, _id field is selected as Shard Key and data block size is set to 1M. See Table 1 for data distribution of scope and Hash-based sharding:

Table 1 Data Distribution of Range and Hash Based Partitionning

	10000		50000		100000		500000		1000000	
	Range	Hash	Range	Hash	Range	Hash	Range	Hash	Range	Hash
Shard1	5267	3352	45267	16613	18936	33193	155731	166254	520625	333058
Shard2	0	3345	0	16542	81064	33634	0	166925	382979	333686
Shard3	4733	3303	4733	16845	0	33173	344269	166821	96396	333256

From the above table, scope sharding results in uneven distribution of data. Under small data size, certain node will not be stored with data. Hash-based sharding can guarantee basically balanced distribution for data in each node. The data size in each node is not completely same, but the difference is constantly controlled within certain scope. Hash-based sharding is also based on scope, and just hashes the specified shard key to long integer type as final shard key.

3.2 Spatial query for laser-point cloud data

In recent years, mobile terminal popularizes rapidly, and application of LBS-Location Based Service has been more and more widely, for example, to search nearby person or article (restaurant, hotel, movie theater, KTV), and how to dispose geographic location information in LBS application has been a critical technology issue [10]. MongoDB native range supports spatial index and can be directly applied to calculate and query location distance, and shows obvious advantages in disposing geographic location information.

The geographic space index for MongoDB includes 2d and 2dsphere index, which will be used to query points in plane and sphere respectively [11]. 2d index is used in this thesis to conduct spatial query for 1w, 5w, 10w, 50w, 100w laser-point cloud data, and geoNear function is used to query nearest 10 points away from the first point in distance set. See Figure 2 for comparison of geoNear query time for scope and Hash-based sharding.

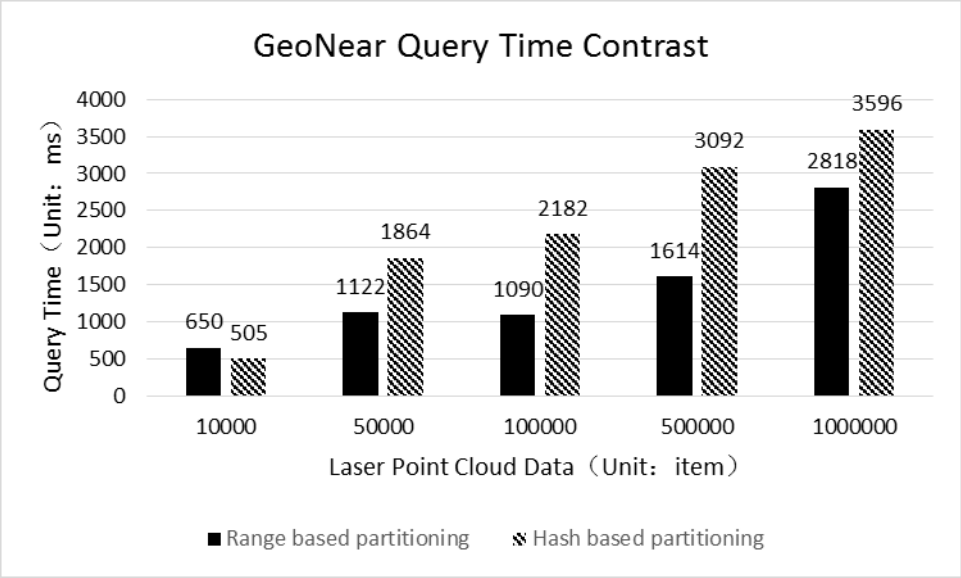

Fig. 2 GeoNear Query Time Contrast of Range and Hash Based Partitioning

From the above figure, sharding cluster can conduct spatial query for millions of laser-point cloud data within several seconds, indicating obvious advantages in disposing geographical location information. MongoDB converts longitude and latitude information to sortable and comparable character string codes through Geohash algorithm [12], and applies common B+ Tree index structure to improve spatial query efficiency. The spatial query efficiency is closely related to laser-point cloud data. The distribution of data in Hash-based sharding is more uniformly, due to increase of data size, the query time will be in stable and increasing trend. However, uneven data distribution in scope sharding results in longer time for 10w data query than 5w data. Under large data size, the query efficiency for scope sharding is higher than Hash-based sharding.

3.3 MapReduce operation for laser-point cloud data

MapReduce is a calculation model for distributed processing, after decomposition for numerous entered data, several servers are used to conduct parallel operation thus to improve efficiency of data processing [13]. MapReduce framework provided by MongoDB is a strong data aggregation tool, equivalent to Group By in relationship data, but Map and Reduce function shall be realized. In this thesis, MapReduce operation has been conducted for 1w, 5w, 10w, 50w and 100w laser-point cloud data, and corresponding data size for single integer in interval [0, 255] has been calculated for R attribute. See Figure 3 for script language JavaScript:

```
mongos> db.runCommand(
... { mapreduce: "p10000",
... map: function () { emit(this.R, 1); },
... reduce: function (key, values) { return Array.sum(values); },
... out: "mrsp10000",
... sort: {R: 1} })
```

Fig. 3 The Scripting Language of MapReduce Operation

The parameters mapreduce and out values respectively represent operable object set and storage set for statistical result. map function invokes emit (key, value) to traverse all records in set, and transfer output key and value to reduce function for processing. The parameter sort refers to sort for objective records, and sorting has certain influence on operation efficiency. See Figure 4 for MapReduce operation time comparison for laser-point cloud data in Hash-based sharding.

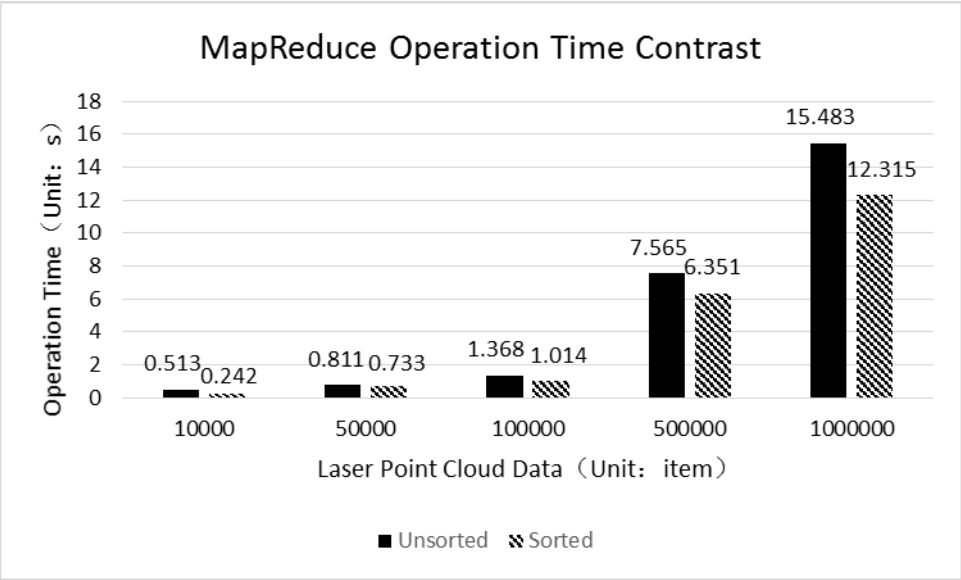

Fig. 4 MapReduce Operation Time Contrast of Unsorted and Sorted

From the above figure, sharding cluster can realize MapReduce operation for millions of laser-point cloud data within several seconds, indicating high-efficient processing of MongoDB for mass laser-point cloud data under distributed environment. The sorting for point cloud data can improve operation efficiency of MapReduce to some extent, and optimize performance for data analysis. If the entered data without sorting are processed, there is almost no chance for MR engine to conduct reduce operation in RAM, which can only write data back to disk through a temporary set, read in order and conduct reduce operation.

4 Conclusions

The laser-point cloud data includes rich geographic information, belonging to scope of spatial data. The application of non-relationship database under distributed environment has provided solution for high-efficient storage and rapid processing of mass spatial data. Sharding technology is a solution for database level extension. In this thesis, sharding cluster for MongoDB is established under distributed environment, while distributed storage, spatial query and MapReduce operation test for numerous laser-point cloud data will be implemented through scope sharding and Hash-based sharding, which completely reflects huge advantages of MongoDB under distribution in storage and processing for spatial data. The next step is to, through machine learning algorithm, combine with high-efficient storage advantage of MongoDB and strong parallel computing power for Hadoop and excavate practical value and social value of mass laser-point cloud data.

References

[1] Zhang Rui, Li Guangyun, Wang Li, et al. *Research on sharding storage method for mass laser-point cloud data based on HDFS* [J]. Bulletin of Surveying and Mapping, 2014(3):21-24.

[2] Chen Jinwei. *Research on key technology of spatial database based on MySQL* [D]. Jiangsu: Nanjing University of Posts and Telecommunication, 2013.

[3] Lu Donghai, He Xianbo. *Brief analysis on NoSQL database* [J]. Science and Technology of West China, 2011, 10(2):14-16.

[4] Wang Guanglei. *Application study and solution optimization for MongoDB database* [J]. China Science and Technology Information, 2011(20):93-96.

[5] Zhang En, Zhang Guangdi, Lan En. *Mass spatial data storage and parallel based on MongoDB* [J]. Geospatial Information, 2014, 12(1):46-49.

[6] Feng Dahui. *Sharding technology for open source database* [J]. Programmer, 2008(7):92-93.

[7] Zhou Wei. *Research on elevating MongoDB auto sharding performance under cloud environment* [J]. Science and Technology Innovation Herald, 2013(29):22-23.

[8] Liang Hai. *Research on sharding technology application in MongoDB database* [J]. Computer Technology and Development, 2014, 24(7):60-67.

[9] Yao Lin, Zhang Yongku. *Distributed storage and extension solution for NoSQL* [J]. Computer Engineering, 2012, 38(6):40-42.

[10] Zhuang Yizhong. *Design and realization based on LBS mobile service framework* [D]. Beijing: Beijing University of Posts and Telecommunications, 2013.

[11] Zhang Guangdi. *Research on storage and parallel query technology for mass spatial data under distributed environment* [D]. Jiangxi: Jiangxi University of Science and Technology, 2012.

[12] Jin An, Cheng Chengqi, Song Shuhua, et al. S*urface data region query based on Geohash* [J].Geography and Geo-information Science, 2013, 29(5):31-35.

[13] Li Chenghua, Zhang Xinfang, Jin Hai, et al. *MapReduce: new distributed and parallel computing programming model* [J]. Computer Engineering and Science, 2011, 33(3):129-135.