Cloud Storage Solution for WSN in Internet Innovation Union

Tongrang Fan, Xuan Zhang and Feng Gao
School of Information Science and Technology, Shijiazhuang Tiedao University, Shijiazhuang, 050043, China
Fantr2009@126.com, zhangxuan@outlook.com, f.gao@live.com

Abstract

With the rapid growth of data storage and processing in demand for Internet Innovation Union (IIU), traditional storage systems could hardly meet most of efficient data access requirements. The paper analyses the advantages and characteristics of private cloud system storage and presents a cloud storage solution of sensor data based on the Hadoop cloud computing framework. Contrast to the private cloud system storage and traditional storage model, the cloud storage solution is more suitable for IIU. The stress tests of the cloud storage solution were performed by compiling YCSB workload class based on MapReduce programming model. The results indicated that the cloud storage solution had better data access performance, and well supported the IIU platform.

Keywords: *cloud storage; Hadoop; HBase; MapReduce; IIU*

1. Introduction

Internet Innovation Union (IIU) is under the funding of the Science and Technology Development Center, Ministry of Education, China, which was confederative initiated by universities, internet industry & research institutes and enterprises in accordance with the principle of voluntariness, equality, cooperation [1]. IIU as a remote network "cloud" experiment platform [2] based on the China Education and Research Network CERNET and next-generation Internet (CNGI-CENRET2). The Internet of things (IoT) as a major component of next-generation technologies is the main direction for future network development [3]. The Internet of things is an extension and expansion of the Internet. The IoT extended the Internet client to any object, communication and information exchange between physical objects using wireless sensors. IIU platform contains different types of sensing equipment, such as temperature sensors, humidity sensors, and infrared sensors. Sensor data analysis may make us aware of the current situation and help us discover the latent and valuable information.

The IIU platform provides the service of mass sensor data storage for Internet of Things, meanwhile the cloud computing also provided the provided technical support to process the data. Therefore, a cloud storage solutions for efficient access to a large number of sensor data in the IIU platform was designed by using Hadoop cloud computing framework, which could support storage and computing services.

2. Related Studies

Hadoop was mainly developed by the MapReduce programming model and GFS (Google File System) file system [4]. The framework mainly included the function of data storage and calculation, and the storage function was provided by the Distributed File system (HDFS), but the clustering algorithms function requirements was satisfied by MapReduce programming model [5]. HBase was an open source implementation of Google's Bigtable, similar to Bigtable database using GFS as a file storage system and HBase using HDFS as a file storage system. HBase provided a high reliability of the underlying storage support by using MapReduce to process huge amounts of data in HBase, provides a high performance computing, row-oriented relational database, in HBase is column-oriented model [6].

The study of employing Hadoop framework to solve problem of the Internet of Things sensor data was few, because of the internet of Things was still in preliminary stage. Christine Jardak et al used MapReduce, and HBase distributed database processing and analysis of data in large-scale WSNs [7]. Wen-Yuan et al proposed a new approach to solve large-scale data storage and analyses of the data read and write performance based HBase distributed database, which were further proved by experiments showing good the performance [8]. Szabolcs Rozsnyai et al put forward a cloud-based commercial source architecture to effectively solve the problems caused by the commercial environment with the advent of large data according to the complex business environment, while today's relational database could not meet the tracking, capture, storage and processing requirements of large amounts of data [9].

In this paper, the issue of a large number of sensor data were storaged in cloud was put forward by using high fault tolerance and scalability advantages based on a Hadoop cluster to solve the IIU platform, which could increase the horizontal memory space and coordinate the load of each server. Finally, the test results indicated that the proposed programs in this article possessed greater flexibility and better met the format of a variety of sensors employing to handle large-scale sensor data with good performance.

3. Cloud Storage Design

3.1. Cloud Storage Structure Model

The Hadoop framework as a cloud computing software platform, users could write distributed parallel processing of massive data program, and the HDFS file system had a high-availability failover and throughput characteristics. HDFS as a basis for cloud storage platform was suitable for processing and storage of large amounts of data. Introducing Hadoop cluster into IIU platform made this system with the ability to handle large amounts of data, and large data processing provide for the application of Internet of Things. The IIU platform cloud storage model structure Based on the Hadoop was shown in Figure 1.

The storage model included Application Programming Interface (API) platform access layer, Metadata storage layer, Data processing management layer, Data storage layer and Distributed coordination services layer. The detailed designs of each layer were described below.

API access layer: API interfaces of the platform access layer provide services for upper layer, mainly including the interfaces of receiving the data and requesting data. The variety of sensors in IIU platform sent to the gateway node connected with the IIU platform via a wired or wireless. The gateway to send data to receive data provided by IIU platform API interface platform, the model will be more data submitted to the HBase database. The user requested that the interface was responsible for command of the user query data submitted to the system, and receiving the results returned from the server and sent to the corresponding user.

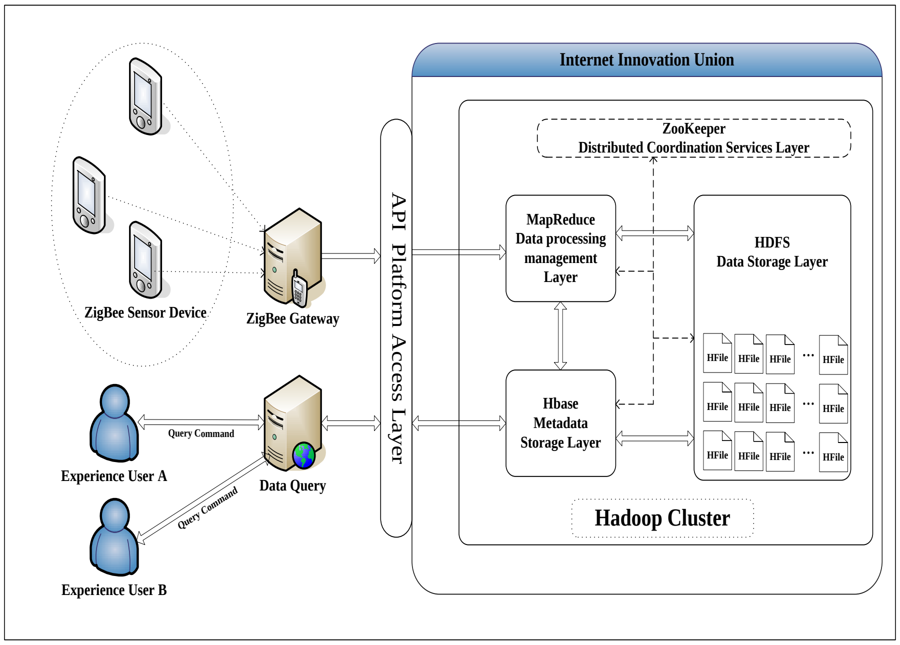

Figure 1. The IIU Platform Cloud Storage Model Structure Based on the Hadoop

Metadata storage layer: The data storage layer is based on HBase, which was also responsible for the creation of data, query and delete operations. Using column-oriented storage could realize scalability and reliability. Creating the metadata process was divided into two phases: the first stage HBase mainly controlling was used to store pre-log file (WAL) and WAL file kept the data changes update operation, including saving, deleting, and inserting et al; the second phase of the main controlling store specific data file will be the actual data deposited in the HDFS. When the ZigBee gateway node had submitted data, simple data judging (to ensuring data validity) was performed. The HBase Client ordered HRegion Server to insert the data requesting command and HRegionServer will match to a specific HRegion. The operation will write into and the data was saved to the WAL MemStore. At the same time MemStore would be checked whether full or not. It would trigger a storage of request to the hard disk, coexistence of written HFile file to HDFS distributed file system.

Data processing management layer: MapReduce computation model data was used to solve complex database operation, including the commands based on user requests on the IIU platform sensor data analysis and processing; A large amounts of data into the database problem, the original input copy the data to HDFS, and then run the MapReduce job, read the input data is written to HBase; and cyclical compression of data in HBase management, distributed database performance could reach optimal situation.

Data storage layer: This object data storage layer was based on the HDFS distributed file system and the data stored in HDFS achieved real sensor data storage and supported for object data storage layer. Meanwhile, data processing management system provided a reliable file storage system.

Distributed coordination services layer: This layer could be achieved by ZooKeeper coordination service framework. ZooKeeper was a reliable coordination system for large-scale distributed systems, which was a key component of Hadoop and HBase. The security, integrity and consistency of the cluster could be ensured By ZooKeeper and furthermore ensuring the stability of the system.

3.2. Sensor Data Model

HBase database data structure design had a direct impact on the performance of database access, and determined the level of data reading performance. According to content of this study, two aspects of sensor data storage structure, namely the row key and the column cluster could improve the designing, so as to meet the needs of the IIU platform, which could been designed as follows:

1) Row key designing

The index key of HBase mainly included Row-key, Column-key and Timestamp. The optimal access speed could be achieved by improving column key designing. Since the time characteristic was the primary characteristic of the sensor data, row key designing could been improve by using the ID of the sensor and receiving time, the designing expression could been described as follows:

<RowKey>: [SensorID, ReceiveTime]

By the definition of the row key, the data and query requirements could be met. The Sensor specific data could be defined by column family.

2) Column designing

For the column stored in HBase, the rows of data supported any number of columns, and the same line allowed a different number of columns. Therefor different column designing had different impact on database performance. The design of column clusters needed to consider many factors, which was more complex. The researchers found that the time-out phenomenon occurred when the number of columns was over 1000. Compared the different column reading / writing efficiency, the experimental results showed the quantitative impact of data was written to the column efficiency.

Sensor data provided important data information and each of sensors was independent of the monitoring data. Once the data deposited into to HBase, it was impossible. Too many column design would increase the frequency of data stored in the database to HBase had an adverse impact would operational efficiency, so the data into a column clusters to reduce the IO frequency of HBase. Therefore, cloud storage solutions in the sensor data structure had been designed as a single cluster of columns "Info" in this article, the designing expression could been described as follows as follows

<Column: Info >: [Info: Temperature, Info: Humidity, Info: Location]

4. Cloud Storage System Performance Testing

Yahoo! Cloud Serving Benchmark (YCSB) is a common performance testing tool of Yahoo's open source. It is implemented by using the Java language test online database performance scalability framework.

4.1. The YSCB Testing Code

Database design in this paper is mainly used to store the received sensor data. Once the data are stored in HBase database, the operation of the data is mainly read. So the level of the performance which a distributed database to read the data have the greatest impact on overall system performance. Because of that the own workload test file in YSCB system only have a basic test to the read, write and update of database. This requires write workload class to meet the data in the MapReduce way to read test. Create new workload class is the in-depth customization for YCSB stress testing model. Develop a set of Extension of the Java file which can be on the HBase database MapReduce stress testing. In order to achieve this goal, the database interface layer and the workload execution layer in YCSB system structure need to make the appropriate changes. Ultimately, it is interacting through this layer when the database interface layer interacts with HBase database, such as read, update, delete and insert operations. Workload execution layer is used to generate the workload class. Through method call com.yahoo.ycsb.DB to have the operations of the storage service finally. The execution of the workload class Corresponds to abstract class: com.yahoo.ycsb.Workload.

In this paper, a distributed storage system focused on the performance of database read and inserts operations. So designs are in two ways: read and insert. In order to meet the needs of the test experiment, the specific design is the following two steps:

(1) The design of new workload class

Abstract class such as yahoo.ycsb.Workload is the basic class of all workload class. So the new workload class MapReduce Workload must inherit from this basic class. At the same time, the new class must have a public constructor with no arguments.

First to increase two abstract function doMapReduceRead() and doMapReduceInsert() in base class of Workload.java file. By using MapReduce method to read and write data in HBase database respectively to do performance testing of the MR model in distributed database.

(2) Design the MapReduce reading / writing class

Described above shows that ultimately interact with the database through the database interface layer. The DB class is the abstract class of all ultimately interacts with the database. So, the design of MapReduce Reading / Writing class MapeduceDB need to inherit the DB cluster. Database read and write functions in MapReduceDB class were MapReduceRead() and MapReduceInsert() functions. The following pseudo code is used to analyze the idea of reading and writing functions respectively:

1) MapReduceRead(): According to the table name and row key passed by doMapReduceRead() function, read the corresponding data in HBase. Pseudo code as follows:

```
BEGIN
Receiving the parameters from doMapReduceRead() function transmission, HTable instance “table” and sensor “ID”;
While (true) {
Using get () function to pass all contents in defined column family INFO_COLUMNFAMILY;
According to the data line, the operation needs the column of cells to obtain the value of the cell;
Access the information such as Location data, Sensor data Etc;
Transferring information obtained to the ResultMap;
Returns the results obtained to ResultMap;
Run next Map processing;
}
END
```

2) MapReduceInsert(): This Methods of data inserted are mainly according to the parameter passed by doMapReduceInsert()method. Pseudo code as follows：

```
BEGIN
Received the parameters passed by function of doMapReduceRead; Insert the line values of Key and the sensor data value;
Receive transmission parameter from the function of doMapReduceInsert();
While (true) {
To judge the legitimacy of the Value data, continue execution if it is legitimate, otherwise, end of this cycle;
Use rows which ready to be inserted and the received time as a line key RowKey;
Use RowKey creating Put object p;
Calling Put add() function to get the inserted position of column clusters, columns, and to be inserted value;
Using the table's put() function to get p inserted into the HBase database;
Run next Map processing;
}
END
```

Through the above two steps, create a new testing class to complete and preparation, then you could use MapReduce-based Workload class to test HBase database performance.

4.2. Testing the Performance of Database

The entire testing process was divided into two parts: the first part was to test storage system performance changes for reading the data; the second part was to test changes for inserting the data. The stress testing storage system performance for reading and inserting experiment were conducted by MapReduce model class in Section 4.1. In two kinds of test, the runtime, throughput and average latency were recorded in the capacity of one million, two million, three million and four million. It may result some errors due to the network environment, but the error was minimized through several paralleling tests.

Experimental environment of the cloud storage systems were shown as follow: hardware information for the four 6-core processor, 6G memory, 2T hard disk server. HBase database was configured as in Section 3.2. Using YCSB workload class loading data to the “usertable” table and could make good prepare for the performance testing data.

4.2.1. Reading Data Performance Analysis: After sensor data stored in the database, the mainly job was to read and analyze the data records stored in the database in future. Reading data performance of the cloud storage system was mainly considering the target of the text storage solutions. The runtime, throughput and latency times were shown in Figure 2, 3 and 4 respectively. Some obvious result could be obtained from these figures. (1) The runtime showed gradually increasing trend with the number of user concurrent increasing in the same scale. (2) The runtime showed gradually increasing trend with database size increasing for the same user number. The longest time was 8.5 second for 4 million records in the Fig.2.The minimum throughput was 1166 ops/sec and the average latency was 106 ms, which were shown in Figure 3 and 4 respectively. The results showed that the HBase database could fully meet requirements for the large number of sensor of reading data application by comparing the runtime, throughput and latency times.

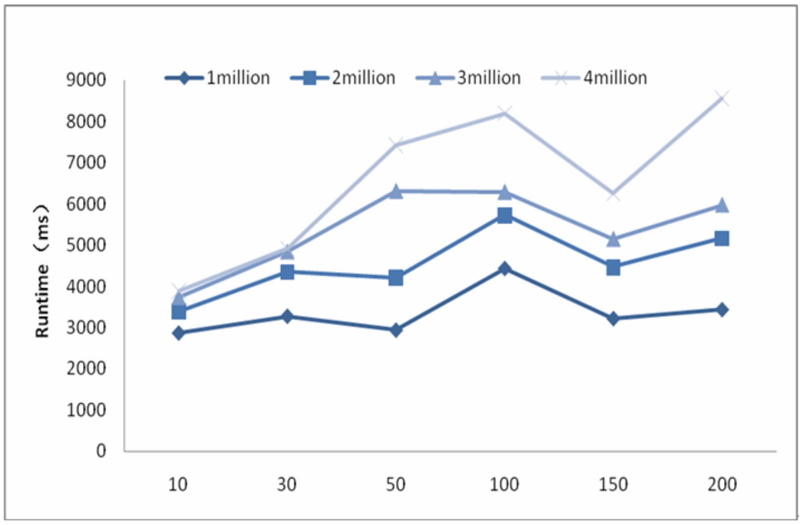

Figure 2. The Runtime for Reading 10000 Records

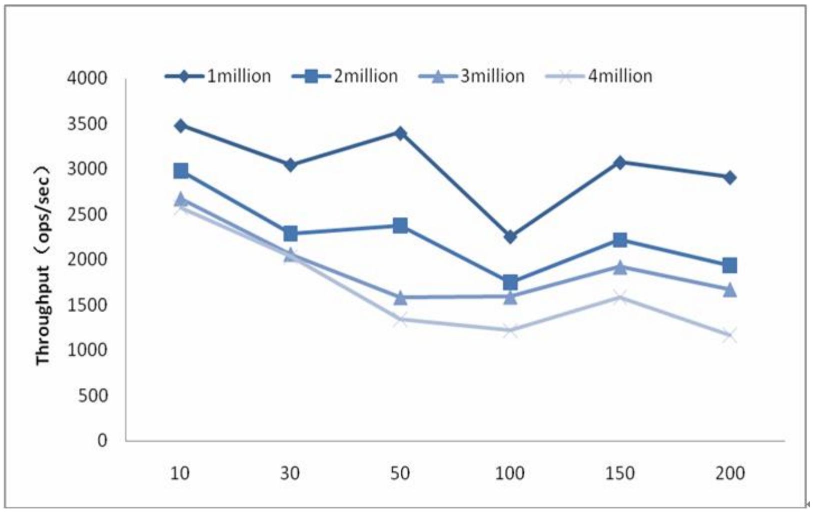

Figure 3. The Throughput for Reading 10000 Records

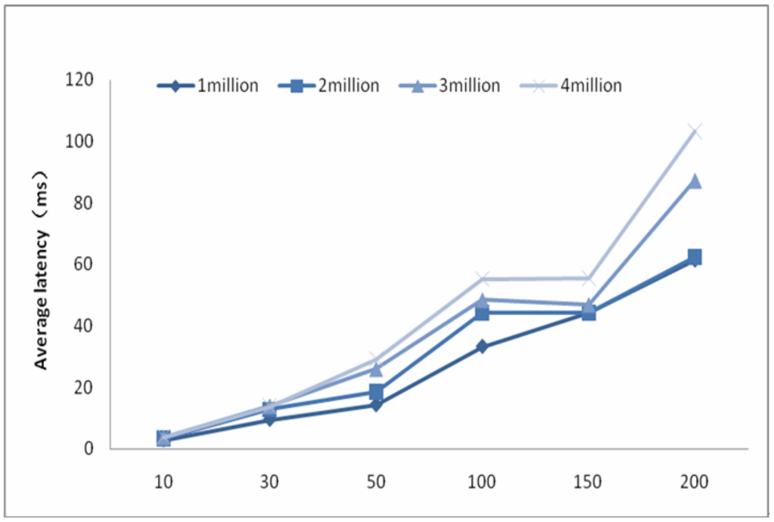

Figure 4. The Average Latency for Reading 10000 Records

4.2.2. Inserting Data Performance Analysis: Importing a large number of sensor observations data to the database was an extremely complex process, which needed long-running database operations. MapReduce and of HBase the distributed model can make full use of the cluster and copy original input data to the HDFS, then run the MapReduce job and read the input data and writes to HBase. The performance of the database into a data cloud storage system solutions had a significant impact on program design. The inserting data performance of configuring system was tested by MapReduce Workload in Section 4.1. The inserting time will increase if the inserting location was different from the original position, just like reading data process. The inserting time will increase with the user increasing in the 1, 2, 3, 4 million scale. The longest inserting time was 14 sec in the 4 million scales. The graphs of throughput and latency time were shown in Figure 6 and 7. The minimum throughput was 788 ops/sec and the average latency was 50 ms, which were shown in Figure 6 and 7 respectively. The higher perform of inserting data still kept with the database scale increasing, which was able to meet our experimental requirements.

5. Conclusions

A Cloud storage module designing in IIU platform based on the environment was proposed, which used the MapReduce programming model YCSB wrokload class testing. By testing data reading and writing performance of the designed cloud storage, the experimental results showed that Hadoop cluster of cloud-based storage performance displayed stable performance with the increase in the amount of data changes by comparing run time, throughput and average latency, the designed IIU platform cloud storage solutions met the experimental application requirements. Cloud storage supporting module designed to provide specific services for the future Internet of Things applications, such as data storage, data analysis and management services to supporting and expanding the scope of the IIU platform. Due to hardware environmental constraints, the cluster data reading and writing performance would perform better when the number of nodes in the cluster increased.

Acknowledgements

This work is supported by Information Fusion and Cloud Services Network Platform project, which is a part of Internet Innovation Union (IIU).(Minister of Education, Science and Technology Development Center[2012]32). The research is also funded by Could Resource Management in Dragon-lab Project (ZH2011102).

References

[1] I. Foster, Y. Zhao, I. Raicu and S. Lu, "Cloud computing and grid computing 360-degree compared", Grid Computing Environments Workshop, GCE 2008, Austin, TX, United States, **(2008)** November 12-16.

[2] J. -L. Wang, C. S. Ni, J. P. Wu and Y. F. Guo, "Next generation internet", Proceedings of the International Symposium on Parallel Architectures, Algorithms and Networks, Hong Kong, China, **(2004)** May 10-12.

[3] F. Guerriero, V. Antonio, N. Enrico L. Valeria and C. Carmelo, "Modelling and solving optimal placement problems in wirelesssensor networks", Applied Mathematical Modelling, vol. 35, **(2011)**, pp. 230-241.

[4] N. Maheshwari, R. Nanduri and V. Varma, "Dynamic energy efficient data placement and cluster reconfiguration algorithm for MapReduce framework", Future Generation Computer Systems, vol. 28, no. 1, **(2012)**, pp. 119-127.

[5] S. Sehgal, M. Erdelyi, A. Merzky and S. Jha, "Understanding application-level interoperability: Scaling-out MapReduce over high-performance grids and clouds", Future Generation Computer Systems, vol. 27, no. 5, **(2011)**, pp. 590-599.

[6] S. N. Srirama, P. Jakovits and E. Vainikko, "Adapting scientific computing problems to clouds using MapReduce", Future Generation Computer Systems, vol. 28, no. 1, **(2012)**, pp. 184-192.

[7] C. Jardak, J. Riihijärvi, F. Oldewurtel and P. Mähönen, "Parallel processing of data from very large-scale wireless sensor networks", Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing, Chicago, IL, United states, **(2010)** June 21-25.

[8] W. Y. Ku, T. Y. Chou and L. K. Chung, "The Cloud-Based Sensor Data Warehouse", International Symposium on Grids and Clouds and the Open Grid Forum, Taipei, Taiwan, **(2011)** February 21-24.

[9] S. Rozsnyai, A. Slominski and Y. Doganata, "Large-Scale Distributed Storage System for Business Provenance", Proceedings-2011 IEEE 4th International Conference on Cloud Computing, CLOUD 2011, Washington, DC, United states, **(2011)** July 4-9.

[10] J. Woo, "Market Basket Analysis Algorithm on Map/Reduce in AWS EC2", International Journal of Advanced Science and Technology, vol. 46, **(2012)**, pp. 25-38.

[11] A. Rehman and M. Hussain, "Efficient Cloud Data Confidentiality for DaaS", International Journal of Advanced Science and Technology, vol. 35, **(2011)**, pp. 1-10.

[12] S. Abdi and S. M. Hashemi, "Job Scheduling and Data Replication in Hierarchical Data Grid", International Journal of Advanced Science and Technology, vol. 47, **(2012)**, pp. 91-100.

Authors

Tongrang Fan, Doctor, Professor, Master Tutor, assistant dean of School of Information Science and Technology, Shijiazhuang Tiedao University, Shijiazhuang, China. Her major field of study are network technology and network security, Information processing and Network and education.

Xuan Zhang, Graduate student of School of Information Science and Technology, Shijiazhuang Tiedao University, Shijiazhuang, China. His major field of study is the Internet of things technology.

Feng Gao, Doctor, Lecturer, teacher of School of Information Science and Technology, Shijiazhuang Tiedao University, Shijiazhuang, China. His major field of study are Cross-platform mobile development technology, the Internet of things technology.