International Journal of Informatics and Information System ISSN 2579-7069
Vol. 3, No. 3, December 2020, pp. 128-135

A High-Performance Data Accessing and Processing System for Campus Real-time Power Usage

Sheng-Cang Chou [1,*], Chao-Tung Yang [2]

[1,2] Dept. of Computer Science, Tunghai University,Taichung City, Taiwan
[1]g04350028@thu.edu.tw*; [2] ctyang@thu.edu.tw;
* corresponding author

Abstract

With the flourishing of Internet of Things (IoT) technology, ubiquitous power data can be linked to the Internet and be analyzed for real-time monitoring requirements. Numerous power data would be accumulated to even Tera-byte level as the time goes. To approach a real-time power monitoring platform on them, an efficient and novel implementation techniques has been developed and formed to be the kernel material of this thesis. Based on the integration of multiple software subsystems in a layered manner, the proposed power-monitoring platform has been established and is composed of Ubuntu (as operating system), Hadoop (as storage subsystem), Hive (as data warehouse), and the Spark MLlib (as data analytics) from bottom to top. The generic power-data source is provided by the so-called smart meters equipped inside factories located in an enterprise practically. The data collection and storage are handled by the Hadoop subsystem and the data ingestion to Hive data warehouse is conducted by the Spark unit. On the aspect of system verification, under single-record query, these software modules: HiveQL and Impala SQL had been tested in terms of query-response efficiency. And for the performance exploration on the full-table query function. The relevant experiments have been conducted on the same software modules as well. The kernel contributions of this research work can be highlighted by two parts: the details of building an efficient real-time power-monitoring platform, and the relevant query-response efficiency for reference.

Keywords: Internet of Things; Big data warehouse; Real-time processing; Spark; Hive; Impala;

1. Introduction

With the concept of Internet of Things and smart grid is getting popular, people is getting focus on the management of electrical equipment and energy consumption situation as well. However, in the process of generating data from electrical equipment, the volume of data must be increasing. When the amount of data is expanding rapidly, the effectiveness of the traditional data storage system will be challenged. In order to store raw data, long-term historical data, and processed data to facilitate future analysis of electricity consumption behavior, we used the concept of huge data warehouse system to increase system scalability, data integrity, and the establish Hive big data warehouse system. In terms of data querying, we choose impala as SQL engine for processing the data stored in Hive and then compare query efficiency between HiveQL and Impala in different parameter.

2. Related Works

2.1. Background - The Internet of Things (IoT)

First, confirm that you have the correct template for your paper size. This template has been tailored for output on the A4 paper size. If you are using US letter-sized paper, please close this file and download the file "MSW_USltr_format".

2.2. Background - Hadoop

Apache Hadoop [3] is an open-source software framework that being broadly used for big data processing nowadays. It came from the Google File System paper which was published in October 2003 and the paper of MapReduce. The Apache Hadoop framework is built on top of the Hadoop Distributed File System (HDFS), which supports a stable and automatic distributed processing system. Hadoop implements MapReduce programming framework which divided file into the same block size. It allows data fragments parallelly execute on any node in the cluster. Hadoop is designed to provide parallel computing and scale up the processing ability from single server to thousands of machines.

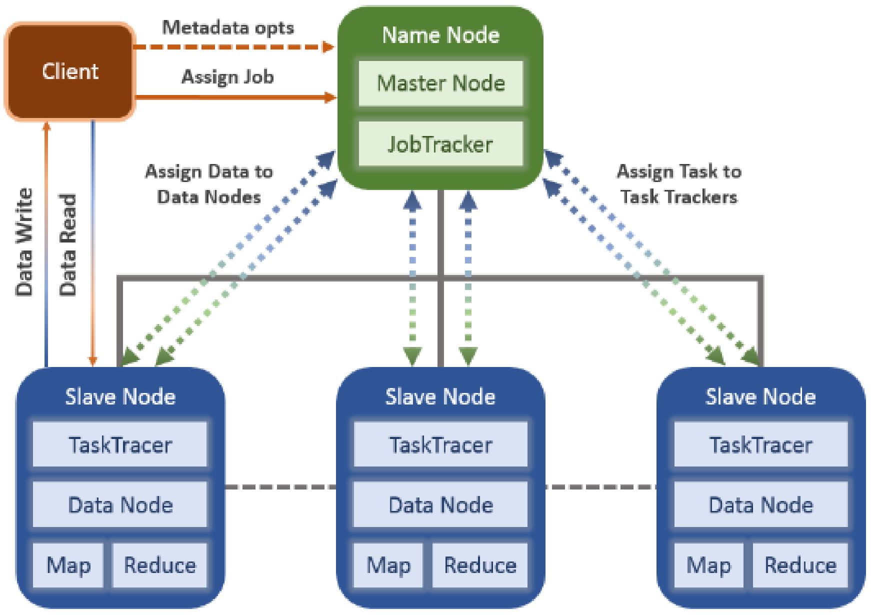

Figure. 1. Hadoop workflow

2.3. Background - Spark

Apache Spark is an open-source cluster computing framework originally developed in the AMPLab at UC Berkeley. In contrast to the two-stage disk-based MapReduce paradigm of Hadoop, Spark in-memory primitives provide performance up to 100 times faster for certain applications. By allowing user programs to load data into a memory of cluster and query it repeatedly, Spark is well suited to machine learning algorithms. Spark requires a cluster manager and a distributed storage system. For cluster management, Spark supports standalone (native Spark cluster), Hadoop YARN, or Apache Mesos.

For distributed storage, Spark can interface with a wide variety, including Hadoop Distributed File System (HDFS), Cassandra, OpenStack Swift, and Amazon S3. Spark also supports a pseudo distributed local mode, usually used only for development or testing purposes, where distributed storage is not required and the local file system can be used instead; in this scenario, Spark is running on a single machine with one executor per CPU core. Spark has over 465 contributors in 2014, making it the most active project in the Apache Software Foundation and among Big Data open source projects

2.4. Background - Impala

Cloudera Impala is a real-time SQL query engine that brings scalable parallel database technology for the Hadoop ecosystem. Itallows user use SQL to query Petabytes of data stored in HDFS and HBase without data movement or transformation. Impala uses Hive metastore, and it can be used to query data from Hive tables directly. Unlike Hive, Impala does not translate the queries into MapReduce jobs but executes them natively.

However, Impala is memory intensive and does not run effectively for heavy data operations like joins because it is not possible to push in everything into the memory. The role of Impala played in Cloudera environment as shown in Figure 2.

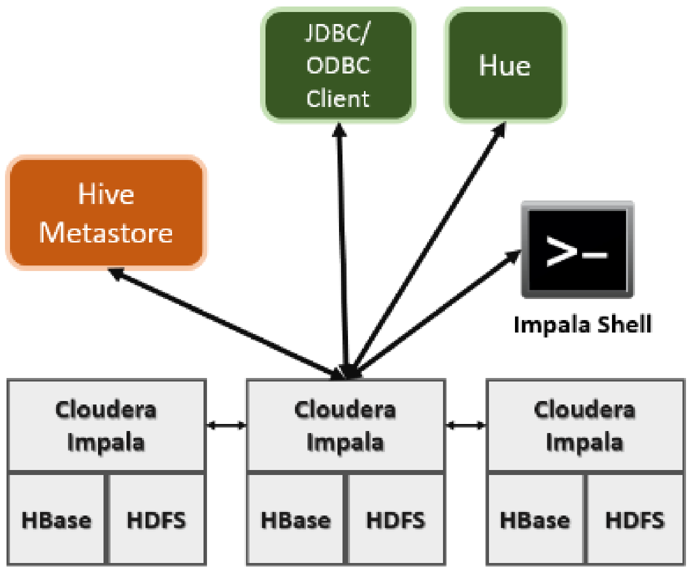

Figure. 2. Impala in Cloudera Distribution Hadoop environment

2.5. Related Works

Smart meter data are typically bundled with social economic data in analytics, such as meter geographic locations, weather conditions and user information, which makes the data sets very sizable and the analytics complex. In Xiufeng Liu et al. [10], they proposed a solution to offer an information integration pipeline for ingesting data from smart meters, a scalable platform for processing and mining big data sets, and a web portal for visualizing analytics results. The implemented system has a hybrid architecture of using Spark or Hive for big data processing, and using the machine learning toolkit, MADlib, for doing in-database data analytics in PostgreSQL database.

Extract-Transform-Load (ETL) tools are pieces of software responsible for the extraction of data from several sources, its cleansing, customization, reformatting, integration, and insertion into a data warehouse. Building the ETL process is potentially one of the biggest tasks of building a warehouse; In Shaker H. Ali El-Sappagh et al. [11], they purposed a model which can be used to design ETL scenarios, and document, customize, and simplify the tracing of the mapping between the data source attributes and its corresponding in the data warehouse.

Through the analysis of OLAP technology in big data environment, a kind of analytical platform of status monitoring big data of electric power equipment was designed. The platform Wang, Dewen, and Zhou. [12] includes relational on-line analysis base on Hive, relational on-line analysis base on Impala, and multi-dimensional on-line analysis based on HBase. Aiming to solve the problems of large cost of connection operation and low query speed of distributed relational analysis data model, they presented a kind of data schema of state monitoring of power equipment which was based on not-join level-encoding technologies. In order to reduce the number of connection options to optimize performance, encoded the level information of dimension table, compressed to the fact table.

3. System Design and Implementation

The experimental system design is important to build a system, we can find out whether the experiment performed is competitive or not based on consider the general design and further action. For more detail, you can see the following experimental environment explication.

3.1. System Architecture

Because the proposed system needs to receive and process power data per second from sensors for energy monitoring, early warning, analysis and other functions, scalability and flexibility of the system are very important. Accordingly, the proposed system has 4-tier architecture, i.e., data generation and collection, data processing and analysis, and data presentation. This 4-tier architecture, as shown in Figure 3, can efficiently process and analyze the huge amount of power data, and its architecture is introduced as follows:

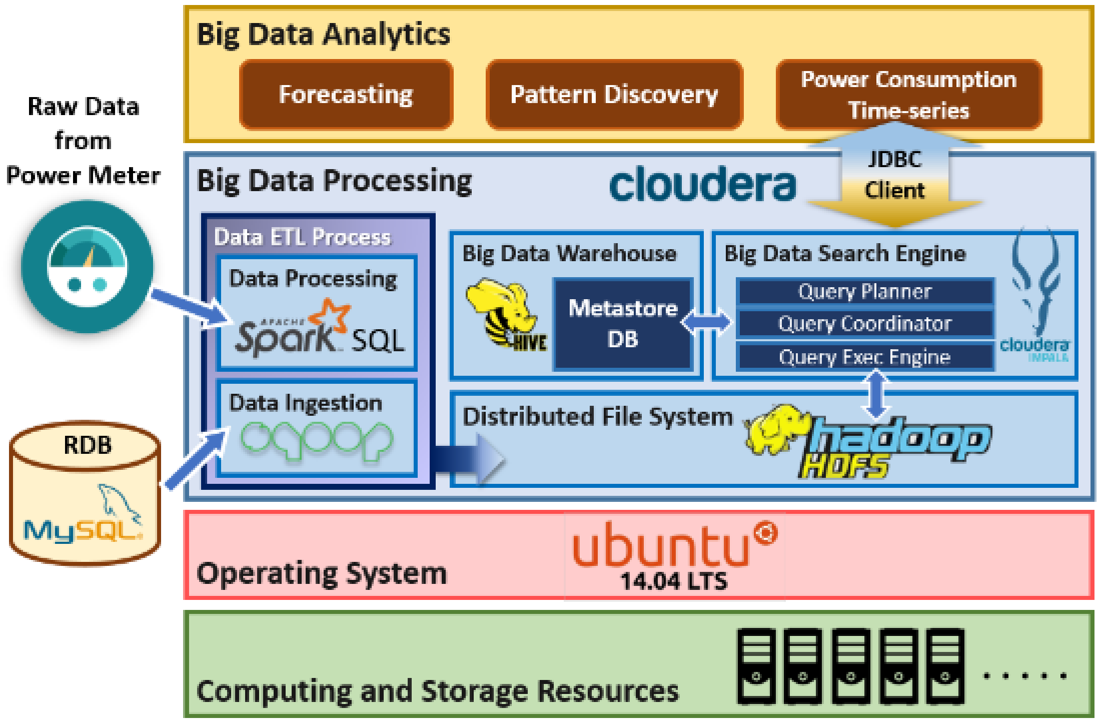

Figure. 3. The overview of system architecture

3.2. Design of Data Warehouse and ETL Service

As seen from the middle layer of Figure 3, Hive became a big data warehouse of the power data and the data source of our system is conducted from the operational MySQL database and real-time smart meter data. The ETL processing of data warehousing had been done with Sqoop as a tool to extract operational databases from MySQL to Hive data warehouse and Spark as real-time data processing engine to transform raw data to useful data. Finally, executing the Spark application periodically to make sure power data can be imported in a stable condition.

3.3. Transferring Operational Data from MySQL to Hive Database

The transfer of legacy meter data from the relational database to HDFS can be done through Apache Sqoop. Sqoop is a data transfer tool that can transfer data from a traditional relational database to a Hadoop storage system by using Hadoop MapReduce parallelism to speed up the process of data migration. It supports not only transfer data from MySQL to HDFS but also Hive and HBase. Figure 4 shows the workflow for operational data transfer.

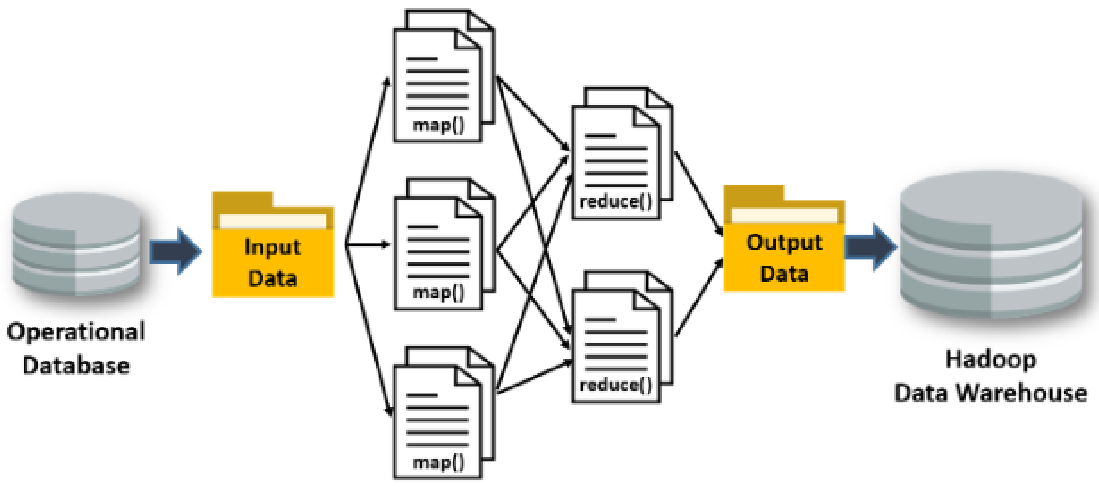

Figure. 4. Sqoop workflow for operational data transfer

3.4. Data ETL Service

Data ETL (Extract-Transform-Load) Service can be used to transfer raw data to the data warehouse via the ETL process. Raw data includes real-time data in data center and data of campus buildings and stable data writing is done by periodically executing Data ETL Service. Figure 5 shows the use case diagram of the data ETL service.

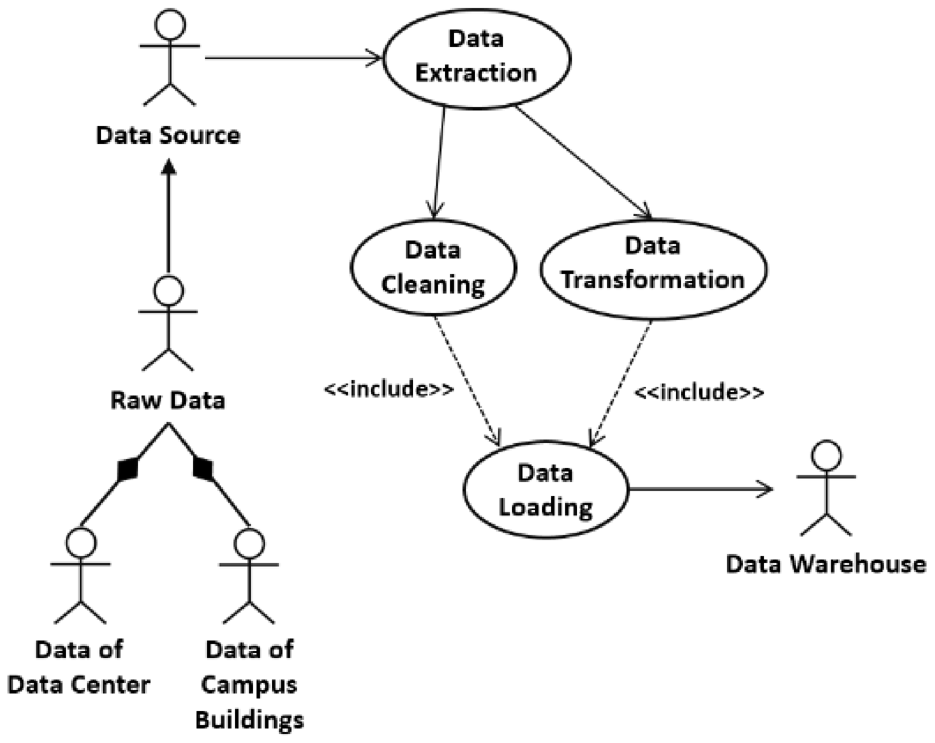

Figure. 5. Use case diagram for data ETL service

3.5. Periodic Statistical Service

To achieve real-time presentation on the front-end web user-interface, the process of calculating raw data to different kinds of processed data periodically is required. By the reason of the efficiency of Spark, scala is our first choose to program the periodic processing application. There are several types of data that is needed to generate, such as accumulated power data in each minute, hour, day, month, and year. Another benefit of adopting spark as back-end processing is to reduce the burden of the front-end.

4. Experimental Results

4.1. Experimental environment

This section presents our hardware and software environmental environment. The proposed system is implemented with 5 physical servers connected by Gigabit Ethernet to build a computing cluster. Each physical server consists of Intel Core i7 CPU with 16 GB Memory and 1TB HD. Besides, Ubuntu 14.05 LTS is adopted as our operating system. Also, the newest version of Hadoop 2.6.0-cdh5.10.1, Spark 1.6.0-cdh5.10.1, Sqoop 1.4.6-cdh5.10.1, Hive 1.1.0-cdh5.10.1, and Impala 2.7.0-cdh5.10.1 in Cloudera Manager 5.10.1 are installed, as shown in Table 1and Figure 6.

Table. 1. Hardware Information

CPU	RAM	HDD	NIC	Number of Core
Intel Core i7-4770	16GB DDR3	1TB	1Gb Ethernet	8
Intel Core i7-4770	16GB DDR3	1TB	1Gb Ethernet	8
Intel Core i7-4770	16GB DDR3	1TB	1Gb Ethernet	8
Intel Core i7-4770	16GB DDR3	1TB	1Gb Ethernet	8
Intel Core i7-6950X	128GB DDR4	1TB	1Gb Ethernet	10

Figure. 6. CDH computing cluster

4.2. Experimental Results

4.2.1. Performance of Full Table Scan among HiveQL and Impala SQL

In the first experiment, we used two kinds of frameworks including HiveQL and Impala SQL to figure out how performance-intensive a full table scan would be on tables with different rows. The range of testing data size is from 217 MB to 9.27 GB. The results show on Figure 7.

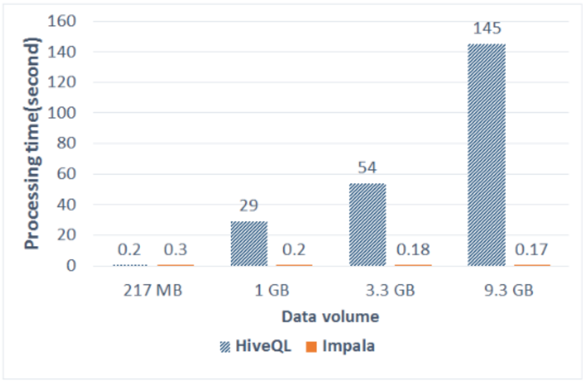

Figure. 7. Comparison of HiveQL and Impala searching speed at full table scan

After the evaluation, Figure 7 shows the result of full table scan. As our expected, the processing time of Impala stays approximately real-time. In the other hand, HiveQL performs as good as Impala when the data size is small, however, when a larger of data volume is given, HiveQL performs substantially worse due to use MapReduce under the hood of Hive

4.2.2. Performance of Querying Single Record between HiveQL and Impala SQL

In order to verify the performance between the three of them in detail, single record querying test has been derived from different scales of data as shown in Figure 8.

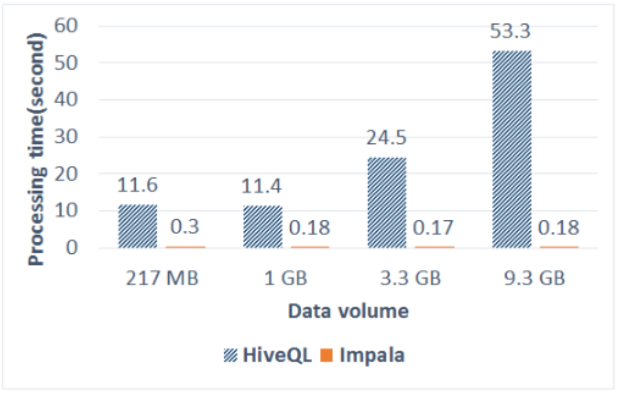

Figure. 8. Comparison of HiveQL and Impala searching speed at single record inquiry

5. Conclusion

5.1. Conclusion

In this work, we built a big data warehouse of power data and ETL processing for data warehousing with several big data modules. The proposed system includes Ubuntu (as operating system), Hadoop (as storage subsystem), and Hive (as data warehouse) from bottom to top. The generic power-data source is provided by the so-called smart meters equipped in real field. The data collection and storage are handled by the Hadoop subsystem and the data ingestion to Hive data warehouse is conducted by the Spark unit. Inorder to evaluate the query-response performance of our data warehouse, several tests had been done in different data volume.

5.2. Future Work

In the future work, we are going to finish the connection between front-end website and the data warehouse that we had built in this work. In that way, the whole real-time power data analytic platform we proposed can be fully achieved. Besides, we will also improve the system by adding different categories of data, such as semi-structure data (like log files) and unstructured data (like video camera recording media). Finally, to confirm the system's scalability we will add more hosts and observe the condition of each hosts to prevent additional incident happened

References

[1] Making hadoop easy with cloudera manager, 2017. https://www.cloudera.com/products/product-components/cloudera-manager.html.

[2] Diane J Skiba. The internet of things (iot). Nursing education perspectives, 34 (5): 63–64, 2015.

[3] Jens Dittrich and Jorge-arnulfo Quian. Efficient Big Data Processing in Hadoop MapReduce. Proceedings of the VLDB Endowment, 5 (12): 2014–2015, 2012.

[4] Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Ning Zhang, Suresh Antony, Hao Liu, and Raghotham Murthy. Hive -A petabyte scale data warehouse using hadoop. Proceedings -International Conference on Data Engineering, pages 996–1005, 2010.

[5] Farag Azzedin. Towards a scalable HDFS architecture. In Proceedings of the 2013 International Conference on Collaboration Technologies and Systems, CTS 2013, pages 155–161, 2013.

[6] Sahithi Tummalapalli and Venkata rao Machavarapu. Managing mysql cluster data using cloudera impala. Procedia Computer Science, 85: 463–474, 2016.

[7] Matei Zaharia, Mosharaf Chowdhury, Michael J Franklin, Scott Shenker, and Ion Stoica. Spark: cluster computing with working sets. HotCloud, 10: 10–10, 2010.

[8] Spark sql, dataframes and datasets guide, 2016. http://spark.apache.org/ docs / 1.6.0 / sql-programming-guide.html.

[9] K. Li, F. Su, X. Cheng, W. Chen, and K. Meng. The research of performance optimization methods based on impala cluster. pages 336–341, 2016.

[10] Xiufeng Liu and Per Sieverts Nielsen. A hybrid ict-solution for smart meter data analytics. Energy, 115, Part 3: 1710 –1722, 2016. Sustainable Development of Energy, Water and Environment Systems.

[11] Shaker H. Ali El-Sappagh, Abdeltawab. Ahmed Hendawi, and Ali Hamed El Bastawissy. A proposed model for data warehouse {ETL} processes. Journal of King Saud University -Computer and Information Sciences, 23 (2): 91 –104, 2011.

[12] D. Wang and Q. Zhou. A method of distributed on-line analytical processing of status monitoring big data of electric power equipment. Zhongguo Dianji Gongcheng Xuebao / Proceedings of the Chinese Society of Electrical Engineering, 36 (19): 5111–5121, 2016.

[13] I. Mavridis and H. Karatza. Performance evaluation of cloud-based log file analysis with apache hadoop and apache spark. Journal of Systems and Software, 125: 133–151, 2017.