Clustering large data sets using MapReduce and Apache Spark

Yoshikazu Yamamoto[*], Mami Matsuda[†], Yuki Fujimoto[‡], Nobuo Shimizu[§] and Junji Nakano[§]

Abstract — **We analyzed a large data set which is sales data from some supermarket chains of ID'S Co. Ltd. i-code data provided by the Joint Association Study Group of Management Science. The data set provided by 6 files. The total size of them is about 5TB. We used MapReduce applications for joining these data files and the aggregating it for each store. The computation was executed by a map-reduce application written in the Java language. We used k-means clustering procedure included in MLlib which is Spark's scalable machine learning library. We wrote program in the Scala language for using it. The result of clustering can explain the characteristics of store groups.**

Keyword: *Big data; Hadoop; k-means; Java language; Scala language.*

1 Introduction

In this paper, we explain handing a large data set and results of the clustering. The data set, which is provided by the Joint Association Study Group of Management Science, is Frequent Shoppers Program (FSP) data of some supermarket chains of ID'S Co. Ltd. i-code data. The total size of the data set is about 5TB. We use some software technologies for "Big Data" such as Hadoop Distributed File System (HDFS), MapReduce and Apache Spark. The data set is stored in HDFS. MapReduce applications perform the data cleansing, the aggregation and the merge of it. Iterative processes for relatively small data set such as the clustering algorithm are executed by procedures on Apache Spark.

We divided stores in the data set into 10 groups by the k-means algorithm implemented by MapReduce applications and Apache Spark. We explain characteristics of store groups by figures for aggregation results.

2 Data set

The data set which we analyzed are provided by 6 files. ID-POS file has 11,360,403,769 lines, Product Master file has 2,226,167 lines, Classification Master file has 1,138 lines, Store Master file has 975 lines, Member Master file has 6,117,712 lines and Product DNA Master file has 1,261,937 lines. These files can be joined by the value of key variables. The total file size of the data set is 5TB.

3 Hadoop MapReduce and Apache Spark

Apache Hadoop is a framework that allows the distributed processing of large data sets across a cluster of computers. It is designed to scale up from single node to thousands of nodes, and each node offers local computational abilities and storage. The core of Apache Hadoop consists of a storage part known as HDFS, and a processing part called MapReduce. Hadoop splits a huge file into rather large blocks and distributes them across nodes in a cluster. MapReduce is an implementation of the MapReduce

[*] Faculty of Science and Engineering, Tokushima Bunri University, 1314-1 Shido, Sanuki-city, Kagawa, 769-2193, Japan, E-mail: yamamoto@fe.bunri-u.ac.jp, Tel: +81-87-899-7100

[†] Information and Communications Second Headquarters, Telecom Second Division, VSN, Inc., 108-0023, Japan

[‡] Tokushima Bunri University, 769-2193, Japan

[§] The Institute of Statistical Mathematics, 190-8562, Japan

programming model for large scale data processing with a parallel, distributed algorithm on a cluster. A MapReduce program is composed of a `map()` method that performs filtering and sorting (such as sorting students by their first names into queues, and each queue contains same first) and a `reduce()` method that performs a summary operation (such as counting the number of students in each queue to have name frequencies).

Apache Spark is a parallel and distributed processing platform and an open source software product supporting big data handling. It is implemented by the Scala language and has some functions such as Mllib and ML Pipeline for machine learning, GraphX, Spark SQL and Spark Streaming. Furthermore, it has features such as high-speed in-memory computing and the interactive environment suitable for trial and error. Apache Spark has the distributed collection called Resilient Distributed Data set (RDD) to store large data set as elements. It supports two types of operations: transformations, which create a new data set from an existing one, and actions, which return a value to the driver program after running a computation on the data set. We can write programs of Apache Spark by the Scala language, the Java language and the Python language. In this paper, we used the Scala language.

The cluster of our distributed processing environment consists of 15 computers which CDH are installed in. CDH, which is Cloudera's open source platform, is the popular distribution of Hadoop and related projects. CDH delivers the core elements of Hadoop scalable storage and distributed computing along with a Web-based user interface. CDH is Apache-licensed open source and offers unified batch processing, interactive SQL and interactive search, and role-based access controls. We used CDH 5.8.0 which include Apache Hadoop 2.6.0 and Spark 1.6.0.

4 Analysis processes

We performed data handling in following processes on HDFS. MapReduce applications executed the data cleansing of original data files, joining and aggregating them. Spark application grouped stores by k-means algorithm.

4.1 Creating input data for Apache Spark

We checked overlap records and the attribute of each fields as the data cleansing. These tasks were performed by MapReduce applications written by the Java language. They read the comma separated value (CSV) files from HDFS and write results to HDFS in the binary format file called Hadoop SequenceFile. The SequenceFile enables fast reading and simple implementation in MapReduce applications. In this process, we did not find any incorrect record.

The inner-join processing of 6 data files by a MapReduce application can be performed by the value of key variables. However, we were not able to join all of them. For example, we tried to join Product Master data and ID-POS data by a value of the key variable Product Information. The number of values of Product Master data (2,226,167) is less than ID-POS data (2,329,938), then we were not able to completely join these data files. The joining Store Master data and ID-POS data was the same situation. The number of values of key variable Store Information of Store Master data (975) is less than ID-POS data (1,003). We decided not to use incomplete data. The analyzed data set size became about 5 TB.

The input data set for clustering with Apache Spark is sum of price in each store and group of a product sold in the store. A MapReduce application read data files from HDFS, aggregated price for each store and product group and wrote the result to HDFS in the SequenceFile format.

4.2 Clusterring

We used the k-means algorithm included in MLlib of Apache Spark for clustering of stores. To decide the number of clusters, we used the value of Within Set Sum of Squared Errors (WSSSE) which Apache Spark provides. We can reduce the error measure by increasing k which is the number of cluster. In fact, the optimal k is usually one where there is an "elbow" in the WSSSE figure. We decided the number of clusters is 10 for this data set.

The MapReduce application merged the group number of each store into the joined data set. We aggregated the data set according to the product group and store group by the MapReduce application for considering the characteristic of the store group.

5 Characteristic of store groups

Figure 1 shows aggregated total values of the daily sales number of each store group. Most groups are variation along constant values. However, group 5 clearly increases after 2015.

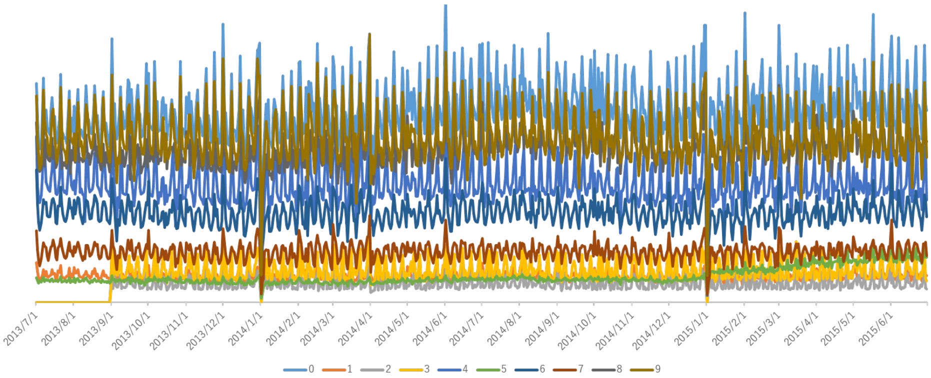

Figure 1: Total values of the daily sales number of each store group

Figure 2 shows the number of sales of the tea-based beverage. As a whole, it has the trend of increasing in summer and decreasing in winter. The group 2 sells most and group 7 is the second in groups.

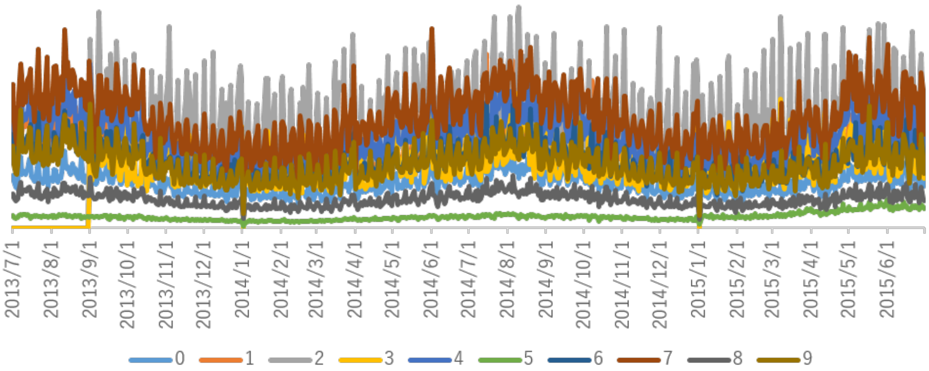

Figure 2: Total values of the daily sales number of the tea-based beverage of each store group

Figure 3 displays the number of sales of the soft drink. In this figure, group 1, which does not appear in figure 1, sells most. Because group 7 emerges in this figure, it sells most both the tea-based beverage and the soft drink. However, group 1 sells a lot of the soft drinks, but it sells little the tea-based beverage.

6 Concluding remarks

We performed the clustering of the large data set on HDFS. MapReduce applications by the Java language created the input data set for clustering algorithm. We used the total value of the sales

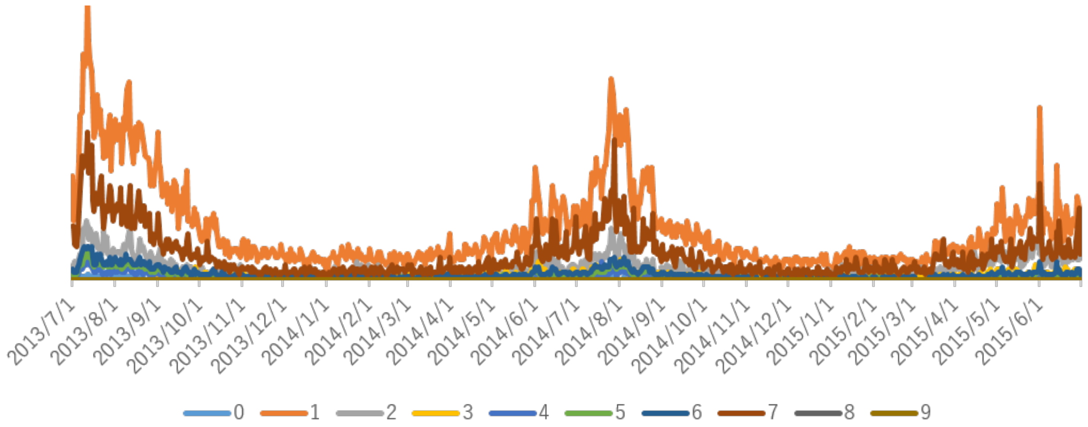

Figure 3: Total values of the daily sales number of the soft drink of each store group

number of product groups of each store. We wrote the program using k-means algorithm for the clustering by the Scala language on Apache Spark. The result was merged into the data set by the MapReduce application. Finally, we aggregated the data set.

We selected 10 store groups by WSSSE values. We aggregated the total value of the daily sales number for every product group of each store group. The store group is characterized by the product. Some groups sold many cigarette and chocolate, and other sold milk and sweet rolls, and some group sold much soft drink.

It is effective for the large data analysis to use both MapReduce and Apache Spark properly well. MapReduce applications can handle very large data sets. We also can use k-means algorithm by the MapReduce application with Apache Mahout, a MapReduce machine learning library. However, it requires a long processing time. We can use high-speed in-memory computing of Apache Spark, if the data set size is smaller than the total memory of nodes in the cluster.

7 Acknowledgment

This work was partly supported by KAKENHI (15K00059; Grant-in-Aid for Scientific Research (C)).

References

[1] Karau H., Konwinski A., Wendell P., Zaharia M. (2015). *Learning Spark: Lightning-Fast Big Data Analysis* O'Reilly Media, Sebastopol

[2] Ryza S., Laserson U., Owen S., Wills J. (2015). *Advanced Analytics with Spark: Patterns for Learning from Data at Scale* O'Reilly Media, Sebastopol

[3] White T. (2012). *Hadoop: The Definitive Guide Third Edition Edition*. O'Reilly Media, Sebastopol