Market Basket Analysis using Spark
Jongwook Woo
Prof., Department of Computer Information Systems, California State University Los Angeles
jwoo5@calstatela.edu

ABSTRACT
Market Basket Analysis (MBA) is one of data mining topics in order to show data association. Apriori algorithm has been popular in sequential computing but Big Data analysis using Hadoop MapReduce has been another approach for large scale data. Recently, Spark has received highlights as it runs on Hadoop Distributed File Systems (HDFS) and much faster than legacy MapReduce. Thus, Market Basket Analysis algorithm in Spark is illustrated with its experimental result in this paper.

Keywords: Hadoop, Big Data, Spark, Market Basket Analysis

1. INTRODUCTION
Big Data becomes common sense these days in order to store and compute large scale data. Data grows exponentially past several years because of social network, sensor networks, bioinformatics, and smartphones, which is called Big Data. Apache Hadoop platform has become a most popular solution to store and process Big Data with the commodity servers, which is not expensive [1].

Recently, Spark becomes a leading computing platform for large scale data analysis as it is much faster using In-Memory processing than the traditional MapReduce while it maintains data locality on HDFS. Spark code can be built in Scala, Java, Python, which are popular programming languages [2].

Market Basket Algorithm is to analyze associates transaction items in order to utilize store stocks, product displays, and item discounts at store. Its algorithm with the experimental result is built in Spark in the paper.

Section 2 describes the existing work. Section 3 illustrates Spark and Market Basket Analysis with its data set. In Section 4, the proposed data analysis algorithm and codes are presented. Section 5 discusses the results of our experimental evaluation. We conclude the research in Section 6.

2. RELATED WORK
Woo et al [3-4] propose Market Basket Analysis algorithm on MapReduce, which run on Amazon AWS.
Woo et al [5] integrates Market Basket Analysis algorithm with NoSQL DB on Amazon AWS.
Apriori-algorithm is presented on MapReduce by Woo [6] but it requires huge memory size.

3. SPARK AND MARKET BASKET ANALYSIS
3.1 Market Basket Analysis [3 – 5]
Market Basket Analysis is one of the Data Mining approaches to analyze the association of data sets. The basic idea is to find the associated pairs of items in a store when there are transaction data sets as in Figure 3.1.
3.2 Spark [2]

Spark is built by AMP Lab of UC Berkeley as a research and teaching project in parallel distributed computing systems. Spark supports Spark Shell for interactive process and the application can be written in Scala, Python, Java. It is now Apache open source project and DataBricks Inc lead the Spark project.

Data for Spark is split out and distributed to nodes that are servers to compose a cluster. The data maintains lineage to track back the history of the data and it can be used to recover when the data is lost in memory. It is called RDD (Resilient Distributed Data). The cluster keeps data on HDFS as well as Mesos that is distributed file systems at UC Berkeley. RDD is a fundamental unit of data to perform an operation. RDD is created from file(s), data in memory, and another RDD.

There are two types of RDD operations: Actions and Transformations. Actions returns values and Transformations defines a new RDD from the existing RDD. RDD is not materialized in memory until an action, which is called lazy execution.

Scala is a functional language that does not keep any state nor side effects. And it supports chain transformation which is a chain of function calls. The chain ends when an action is performed.

Spark is composed of Spark SQL, Spark R, Machine Learning, Graph, and Streaming APIs. Spark Streaming provides almost real time processing of streaming data that is mostly from servers in order to monitor and detect website, malfunctions, intrusion, commercial ad, and fraud.

In summary, Spark provides much faster computing power with In-memory processing than MapReduce and fault tolerance on HDFS maintaining data locality with multiple APIs.

4. MARKET BASKET ANALYSIS USING SPARK

1. Take an input transaction text
2. Items are sliced by n elements and sorted
   a. Slided items are generated as (e1, e2, ..., en)
3. Duplicated items pairs are removed
4. Several (e1, e2, ..., en) pairs are produced by n

Thus, when the transaction is input, the output of Figure 3.4 becomes:

```
[(beer, coke), 1]
[(beer, cracker), 1]
[(coke, cracker), 1]
```

5. EXPERIMENTAL RESULT

The application is built in Spark Scala and processed on Amazon AWS, AWS EC2 m1.large and m1.xlarge. Spark version is 1.3.0 and the data is stored at its HDFS on the cluster. m1.large instantiate a server with 2 core (2 EC2 vCPU compute unit), 7.5GB memory and 2 x 420GB storage on 64 bits platform Amazon Linux version 2015.03 64bits OS. m1.xlarge uses a server with 4 core (4 EC2 vCPU compute unit), 15GB memory and 4 x 420GB storage on 64 bits platform Amazon Linux version 2015.03 64bits OS.

Because of the resource restriction, the number of nodes instantiated on AWS EC2 are 2, 4, and 6 nodes – 1 is a master node - and the input transaction data sets are 1.6 GB and 3.2 GB file size.

1) 1.6GB transaction data set

![Figure 5: Performance for 1.6GB file](image)

Figure 5 shows that the more the nodes the fast the computation time is. At m1.large, 3 and 5 slave nodes are 3.43 ~ 3.72 times much faster than 1 slave node. At m1.xlarge, 3 and 5 slave nodes are 1.93 ~ 3.6 times much faster than 1 slave node. Besides, m1.xlarge has up to 55% better performance than m1.large.

2) 3.2GB transaction data set

![Figure 6: Performance for 3.2GB file](image)

Figure 6 shows that the more the nodes the fast the computation time is. At m1.large, 3 and 5 slave nodes are 3.43 ~ 3.72 times much faster than 1 slave node. At m1.xlarge, 3 and 5 slave nodes are 1.93 ~ 3.6 times much faster than 1 slave node. Besides, m1.xlarge has up to 55% better performance than m1.large.
Figure 4.2 shows that the more the nodes the fast the computation time is. At m1.large, 3 and 5 slave nodes are 3.43 ~ 4.45 times much faster than 1 slave node. At m1.xlarge, 3 and 5 slave nodes are 2.36 ~ 6.32 times much faster than 1 slave node. Besides, m1.xlarge has up to 53% better performance than m1.large. The code will be uploaded as MBA Spark example to https://github.com/hipic.

Figure 7: illustrates the performance of all data on slave nodes 1, 3, and 5 at m1.large and m1.xlarge instances.

6. CONCLUSION
Spark leads Big Data computing community as it is In-Memory processing that achieves much better high performance than the legacy MapReduce processing.

The paper presents an MBA algorithm on functional programming. And, the MBA code is written in Scala Spark and processed on AWS EC2. The experimental results show that the more nodes the better performance is achieved. Besides, the more memory and CPUs the better performance is gained.

REFERENCES

AUTHOR PROFILES
Jongwook Woo is currently a Professor at Computer Information Systems at California State University, Los Angeles. He received the BS and the MS degree, both in Electronic Engineering from Yonsei University in 1989 and 1991, respectively. He obtained his second MS degree in Computer Science and received the PhD degree in Computer Engineering, both from University of Southern California in 1998 and 2001, respectively. His research interests are Information Retrieval /Integration /Sharing on Big Data, Map/Reduce and functional algorithm on Hadoop Parallel/Distributed/Cloud Computing, and n-Tier Architecture application in e-Business, smartphone, social networking and bioinformatics applications. He has published more than 40 peer reviewed conference and journal papers. He also has consulted many entertainment companies in Hollywood.