International Journal of Advances in Computer Science & Its Applications– IJCSIA
Volume 5 : Issue 2 *[ISSN : 2250-3765]*
Publication Date: 30 October, 2015

Review of several improved Apriori algorithms on Hadoop-MapReduce environment

A.L.Sayeth Saabith[1], Elankovan Sundararajan[2], Azuraliza Abu Bakar[3]

***Abstract*— Association Rule Mining (ARM) plays a significant role in the data mining techniques. ARM aims to reveal association relationship among different items in large datasets. The Apriori algorithm is one of the most broadly used algorithm in ARM that collects the item sets which frequently occur in order to discover association rule in massive datasets. The original Apriori algorithm is for the sequential (single node or computer) environment. This Apriori algorithm has many drawbacks to process huge datasets. Many researches have been carried out for parallelizing the Apriori algorithm. This study does a survey on few good improved and revised approaches of parallel Apriori algorithm on Hadoop-MapReduce environment. Hadoop-MapReduce framework is a programming model that efficiently and effectively processing enormous databases in parallel on large clusters of commodity hardware in a reliable, and fault- tolerant manner. This survey will provide an overall view of parallel Apriori algorithm implementation over Hadoop-MapReduce environment and briefly discussing Hadoop challenges and advantages.**

Keywords—*ARM, Apriori, Hadoop, MapReduce.*

I. INTRODUCTION

Data mining is the process of extracting useful, potential, novel, understandable, hidden information from the datasets which are huge, noisy, and ambiguous. Data mining plays a vital role in various application in the modern world such as market analysis, credit assessment, fraud detection, medical discovery, fault diagnosis in production system, hazard forecasting, customer relation and science exploration. Many people treat data mining as a synonym for another popularity used term, Knowledge Discovery from Data (KDD), while others view data mining as merely an essential step in the process of KDD[8]. Overview of the steps constituting the KDD process, as shown in Fig. 1.

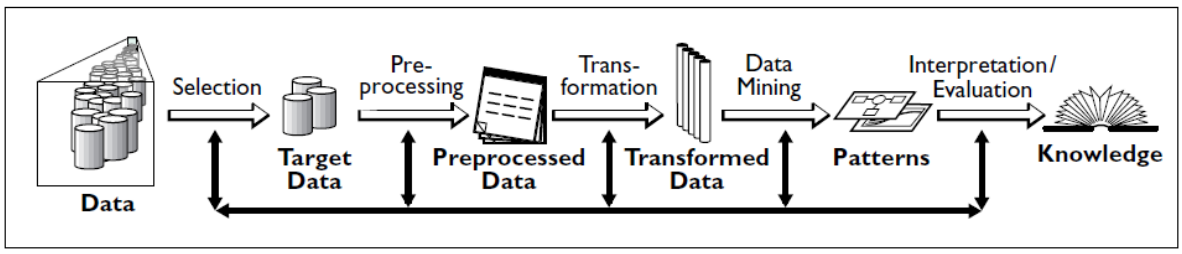

Fig. 1 Knowledge discovery of KDD process [19]

Business intelligence has become an integral part of many successful organizations. Analyzing data and making decisions based upon the analysis is very important for an organization growth. Data mining techniques help analyze the substantial data available to assist in decision making. One of the most important areas of the data mining is Association Rule Mining (ARM) or Frequent Itemset Mining (FIM). ARM intentions to extract interesting correlations, patterns, associations among sets of items in the transaction database or other data repositories[9, 10]. The most typical application of ARM is in market basket analysis which analyzes the purchasing behavior of customers by finding the frequent item purchased together. In addition to the many business application, it is also applicable to telecommunication networks, web log mining, market and risk management, inventory control, bio-informatics, medical diagnosis and text mining[8]. The Apriori algorithm is one of the best classical algorithms for discovering frequent itemsets from a transactional database; but, it has some drawbacks such as it scans the dataset many times to generate frequent itemsets and it generates many candidate itemsets. When data mining mainly deals with large volumes of data, both memory use and computational cost can still be very expensive, also single processor's memory and central processing unit resources are very limited, which make the algorithm performance inefficient[4]. One way of improving the performance and efficiency of Apriori is parallelizing and distributing the process of generating frequent itemsets and association rules. These versions of parallel and distributed Apriori algorithms improve the mining performance but also has some overheads such as workload balancing, partition of input data, reduce the communication costs and aggregation of information at local nodes to form the global information. The problems with most of the distributed framework are overheads of managing distributed system and lack of high level parallel programming language.

Working with large number of computing nodes in cluster or grid, there is always a potential chance of node failures which causes multiple re-executions of tasks. All these pitfalls can be overcome by the MapReduce framework introduced by google[11].

A.L.Sayeth Saabith[1], Elankovan Sundararajan[2], Azuraliza Abu Bakar[3]
[1,2]*Centre for Software Technology and Management*
[3]*Center for Artificial Intelligence and Technology*
Faculty of Information Science and Technology
Universiti Kebangsaan Malaysia, UKM Bangi, 43600, Selangor-DE, Malaysia.

Hadoop-MapReduce model which is a Java based programming model for easily and efficiently writing applications that process vast amount of data in parallel on large clusters of commodity hardware in a reliable, fault-tolerance manner.

In this study, we explore detail review of several improved Apriori algorithm over Hadoop-MapReduce environment. Rest of the paper is organized as follows: The rest of this paper is organized as follows. Section II presents six common classes of data mining algorithms, basic concept of ARM, and Apriori algorithm. Section III provides an overview of parallel discovery of Apriori algorithm. Section IV describes the concepts of Hadoop, MapReduce, and HDFS and how Apriori algorithm implements on Hadoop-MapReduce model with example. Section V presents analysis of several improved Apriori algorithms on Hadoop-MapReduce environment. . Section VI briefly explains the advantages and challenges of Hadoop.

II. DATA MINING ALGORITHMS

A data mining algorithm is a well-defined procedure to create a data mining model from data repositories. Usually data mining algorithm analyzes the data to create a model considering specific types of patterns and trends. The algorithm reveal the results of this analysis to define the optimal parameters for creating the data mining model. These parameters are helping to extract the patterns and detailed statistics from the entire dataset. Data mining algorithms involves six common classes of data mining techniques which are describe as shown in Table. 1.

TABLE. 1 SIX COMMON CLASSES OF DATA MINING ALGORITHM

Category	Description	Algorithms	Application area
Anomaly Detection or Outlier Detection	Anomaly or Outlier detection is refers to the identification of the items, events, and observations in dataset which does not confirm to expected behavior	SVM, Fuzzy Logic, K-Means, Nearest Neighbor, and Outlier Count	Intrusion detection, Fraud detection, Fault detection, System health monitoring, and Event detection in sensor networks.
Association Rule Mining(ARM)	ARM attempts to find frequent itemset among large datasets and describes the association relationship among different attributes.	Apriori, Eclat, FP growth, Partition	Web usage mining, Intrusion detection, Continuous Production, and Bioinformatics
Clustering	Clustering is the process of partitioning a set of objects in such a way that objects in the same group are more similar to each other than to those in other groups	K-Means, DBSCAN Fuzzy C Means, and Expectation Maximization	Machine Learning, Pattern Recognition, Image Analysis, Information Retrieval, Bioinformatics, Crime Analysis, and Climatology
Classification	Classification is the data mining function that assigns items in a collection to target categories or classes. The goal of classification is to accurately predict the target class for each case in the data.	Decision Tree, Naïve Bayes, KNN, GLM, and SVM	Fraud Detection, Credit Risk, Stock Market, DNA, and E-mail
Regression	Regression is commonly used to predict future values base on past values by fitting a set of points to a curve.	Multivariate, and Adaptive Regression	Detect Fraud, and Minimize risk
Summarization	Providing a more compact representation of the data set, including visualization and report generation	LexRank, and TextRank	multimedia document, Text summarization, and Image collection

A. Association Rule Mining (ARM)

ARM is one of the key class of data mining techniques and it was introduced by[9]. The aim of ARM is to extract interesting relationships, frequent patterns, association or casual structures among sets of items in the transaction databases or other data repositories. We elaborate some generic concepts of association rules mining according to[9]. Let $I = \{i_1, i_2, \ldots, i_n\}$ be a set of n distinct literals, called items. Let $D = \{t_1, t_2, \ldots . t_n\}$ be a set of transactions, where each transaction T is a set of items such that $T \subseteq I$. In association analysis, a collection of zero or more items is called an itemset. If an itemset contains k items, it is called k-itemset. Each transaction has a unique identifier TID. The itemset X has support s in the transaction set D if s% transaction contains X, here we called $s = support(X)$. An important property of an itemset is its support count, which is refers to the number of transactions that contain a particular itemset. The support count σ(x) for an itemset X can be state as $\sigma(X) = \{|t_i|X \subseteq t_i, t_i \in T\}$. An association rule is an implication in the form of $X \rightarrow Y, where\ X, Y \subseteq I\ and\ X \cap Y = \phi$.

The strength of association rule can be measured in terms of its support and confidence. $Support\ (S)$ determines how often a rule is applicable to a given dataset. $S(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{N}$. $Confidence(C)$ determines how frequently items in Y appear in transactions that contains X, $C(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}$. The problem of association rule mining is to find all the rules that satisfy a user predefined minimum support (min_sup) and minimum confidence (min_con) . If $support(X)$ is larger than a user defined min_sup then the itemset X is called frequent itemset. The association rule mining task can be decomposed into two sub tasks. The first task is finding all of the frequent itemsets which have support above the user specified minimum support and second task is generating rules from these frequent itemsets[9, 12]. The ARM performance typically depend on the first task. Usually, ARM generates very large number of association rules. Most of the time, it is difficult for users to understand and confirm a large number of complex association rules. So, it is important to generate only "interesting" and "non-redundant" rules, or rules satisfying certain criteria such as easy to handle, control, understand,

and increase the strength. Dozens of algorithms have been developed to find the frequent itemset and association rules in ARM, some of the commonly used algorithms are Apriori algorithm, partition algorithm, pincer search algorithm dynamic item set counting algorithm, FP tree growth algorithm, Eclat and dEclat.

B. Apriori algorithm

Rapid advancement of information technology has resulted in accumulation of tremendous amount of data for organization and therefore extracting needed information from huge amount of data has been a big challenge for researchers[13]. Apriori is classic and broadly used ARM algorithm. It uses an iterative approach called breath-first search to generate $(k-1)$ itemsets from k item sets. The basic principal of this algorithm is that all nonempty subsets of a frequent itemset must be frequent. There are two main steps in Apriori (1). the prune step: remove the itemsets if support is less than min_sup which predefined by user value and discard the itemset if its subset is not frequent. (2). the Join step: the candidates are generated by joining among the frequent item sets level wise. The key drawback of this algorithm is the multiple dataset scan. Fig. 2 describes the pseudo code of the Apriori algorithm.

L_1 =countItems(D)
C_2=generateCandidate(L_1)
i=2
while $C_i \neq \emptyset$ **do**
  for all $t \in D$ **do**
    incrementCounter(C_i, t)
  end for
  L_i=generateFrequents(C_i, min_sup)
  i++
  C_i=generateCandidate(L_{i-1})
end while
return $\bigcup_i L_i$

Fig. 2 Apriori algorithm pseudo code

The input data is usually huge and distributed in nature. Therefore a cloud could be a perfect platform for data mining algorithms. However, the classical Apriori algorithm cannot be implemented in the parallel environment because it was intended for sequential processing. Sequential processing algorithms are impractical to run on cloud environment. So, many sequential and parallel algorithms have been proposed to improve the efficiency and performance of Apriori algorithm.

III. PARALLEL DISCOVERY OF APRIORI ALGORITHM

The constant increase in the volume and detail of the data together with the advent of parallel computing technology, various association mining parallel algorithms have been proposed such as count distribution algorithm, data distribution algorithm, candidate distribution algorithm and improved Apriori algorithm[4, 14, 15]. These parallel algorithms can be implemented under cloud computing environment to reduce computation time, memory usage and I/O overhead for generating frequent item sets. It can boost the performance of association rules mining algorithms. Table. 2 describes the most popular versions of the parallel association rule mining algorithms.

TABLE. 2 PARALLEL APRIORI ALGORITHM TECHNIQUES

Method	Description	Analysis
Count Distribution Algorithm (CDA)	Pass>1 Generate complete C_k from L_{k-1} Count local data D_i to find support for C_k Exchange local count for C_k to find global C_k Compute L_k from C_k Pass k=1 Generate local C_1 from local data D_i Merge local candidate sets to find C_1	Input/output Time $O(N/P)$ CPU Time $O(count)/P + overhead$ Communication Volume $\sum_k O(\lvert C_k \rvert)\ per\ CPU$ Message Count $\sum_k \log P$
Data Distribution Algorithm (DDA)	Partition data + candidate set Generate C_k from L_{k-1} ; retain $\lvert C_k \rvert / P$ locally Count local C_k using both local and remote data Calculate local L_k using the local C_k and synchronize	Input/output Time $O(N/P)$ CPU Time $O(count)/P + overhead$ Communication Volume $\sum_k O(N)\ per\ CPU$ Message Count $\sum_k O(P)$

IV. APRIORI ALGORITHM ON HADOOP AND MAPREDUCE MODEL

A. Hadoop

The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models[16]. It is an Apache project released under Apache Open Source License v2.0. This license is very commercial friendly. Hadoop provides two things: Storage & Compute. If consider Hadoop as a coin, one side is storage and other side is compute. In Hadoop speak, storage is provided by *Hadoop Distributed File System (HDFS)*. Compute is provided by *MapReduce[17]*. Hadoop, as shown in figure, is a distributed master-slave architecture that consists of the

Hadoop Distributed File System (HDFS) for storage and MapReduce for computational capabilities. Fig. 3 describes the high level architecture of Hadoop.

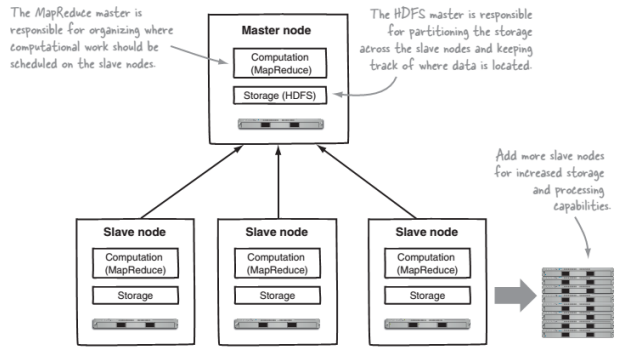

Fig. 3 High-level Hadoop Architecture

B. MapReduce

MapReduce is a batch-based, distributed computing framework modeled after Google's paper on MapReduce[11]. It allows you to parallelize work over a large amount of raw data, such as combining web logs with relational data from an OLTP database to model how users interact with your website. This type of work, which could take days or longer using conventional serial programming techniques, can be reduced down to minutes using MapReduce on a Hadoop cluster. The MapReduce model simplifies parallel processing by abstracting away the complexities involved in working with distributed systems, such as computational parallelization, work distribution, and dealing with unreliable hardware and software. With this abstraction, MapReduce allows the programmer to focus on addressing business needs, rather than getting tangled up in distributed system complications[17, 18]. The power of MapReduce occurs in between the map output and the reduce input, in the shuffle and sort phases, as shown in Fig. 4

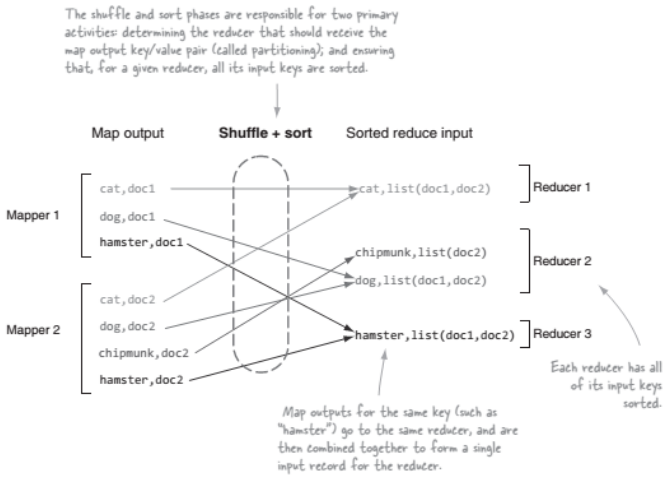

Fig. 4 Map, Shuffle and Reduce process

C. HDFS

HDFS is Hadoop Distributed File System, which is responsible for storing data on the cluster in Hadoop. Files in HDFS are split into blocks before they are stored on the cluster. The typical size of a block is 64MB or 128MB. The blocks belonging to one file are then stored on different nodes. Scalability and availability are also key traits of HDFS, achieved in part due to data replication and fault tolerance. HDFS replicates files for a configured number of times, is tolerant of both software and hardware failure, and automatically re-replicates data blocks on nodes that have failed.

D. Apriori example on Hadoop MapReduce model

The following example briefly describes how to implement Apriori algorithm on Hadoop MapReduce model, as shown in Fig. 5 according to Map, Shuffle, and Reduce process.

Sample dataset

TID	ITEMS
T001	Banana, Juice, Cake, Bread
T002	Banana, Cake
T003	Juice, Cake
T004	Banana, Juice, Bread
T005	Juice, Snack
T006	Banana, Juice, Cake
T007	Juice, Cake
T008	Banana, Juice, Snack
T009	Banana, Cake

Partition the database by three

D1	D2	D3
(T001, {Banana, Juice, Cake, Bread})	(T004, {Banana, Juice, Bread})	(T007, {Juice, Cake})
(T002, {Banana, Cake})	(T005, {Juice, Snack})	(T008, {Banana, Juice, Snack})
(T003, {Juice, Cake})	(T006, {Banana, Juice, Cake})	(T009, {Banana, Cake})

Using the mapper function and getting the

D1	D2	D3
(Banana, 2) , (Juice, 2), (Cake, 3), (Bread, 1)	(Banana, 2), (Juice, 3), (Cake, 1), (Bread, 1), (Snack, 1)	(Banana, 2), (Juice, 2), (Cake, 2), (Snack, 1)

Shuffle and exchange the intermediate value

D1	D2	D3
(Banana, {2,2,2}) , (Snack, {1,1})	(Juice, {2,3,2}, (Bread, {1,1})	(Cake, {3,1,2})

Using the reducer function and generating

D1	D2	D3
(Banana, 6) , (Snack, 2)	(Juice, 7), (Bread, 2)	(Cake, 6)

Fig. 5 Apriori MapReduce Example

V. ANALYSIS OF SEVERAL IMPROVED APRIORI ALGORITHMS ON HADOOP-MAPREDUCE ENVIRONMENT

Table 3 presents several related studies that parallel Apriori algorithm deal with big data through the use of Hadoop and MapReduce distributed framework. The table provides objectives of related papers which are deal with parallel Apriori algorithm based on Hadoop technologies and describes the techniques, and technologies that used in Hadoop MapReduce environment to deal with big data.

VI. ADVANTAGES AND CHALLENGES OF HADOOP

A. Hadoop Advantages

- **Scalable**- Hadoop is a highly scalable storage platform, because it can store and distribute very large data sets across hundreds of inexpensive servers that operate in parallel.
- **Cost effective**- Hadoop also offers a cost effective storage solution for businesses' exploding data sets.
- **Flexible**- Hadoop enables businesses to easily access new data sources and tap into different types of data (both structured and unstructured) to generate value from that data.
- **Fast**- Hadoop's unique storage method is based on a distributed file system that basically 'maps' data wherever it is located on a cluster. The tools for data processing are often on the same servers where the data is located, resulting in much faster data processing.
- **Resilient to failure**-A key advantage of using Hadoop is its fault tolerance. When data is sent to an individual node, that data is also replicated to other nodes in the cluster, which means that in the event of failure, there is another copy available for use.

B. Hadoop Challenges

- **Security Concerns**- Just managing a complex application such as Hadoop can be challenging. A classic example can be seen in the Hadoop security model, which is disabled by default due to sheer complexity.
- **Vulnerable by Nature**- speaking of security, the very makeup of Hadoop makes running it a risky proposition. The framework is written almost entirely in Java, one of the most widely used yet controversial programming languages in existence. Java has been heavily exploited by cybercriminals and as a result, implicated in numerous security breaches.
- **Not Fit for Small Data**- While big data isn't exclusively made for big businesses, not all big data platforms are suited for small data needs. Unfortunately, Hadoop happens to be one of them. Due to its high capacity design, the Hadoop Distributed File System or HDFS, lacks the ability to efficiently support the random reading of small files. As a result, it is not recommended for organizations with small quantities of data.
- **Potential Stability Issues**- Hadoop is an open source platform. That essentially means it is created by the contributions of the many developers who continue to work on the project. While improvements are constantly being made,

TABLE. 3 REVIEW SEVERAL IMPROVED APRIORI ALGORITHM ON HADOOP-MAPREDUCE WITH OBJECTIVE

Reference	Title of the paper	Objective
[7]	"Parallel Implementation of Apriori Algorithm Based on MapReduce"	To evaluate the performance of their proposed Apriori algorithm in terms of size up, speedup, and scale up to deal with large scale dataset
[3]	"An Improved Apriori Algorithm Based On the Boolean Matrix and Hadoop"	Theoretically proved their improved Apriori algorithm. First the Boolean matrix array is used to replace the transaction database: therefore those non-frequent item sets can be removed from the matrix and it does not need to scan the original database; it just need to operate on the Boolean matrix using the vector operation "AND" and the random access characteristics of array so that it can directly generate the k- frequent itemsets.
[4]	"Exploring HADOOP as a Platform for Distributed Association Rule Mining"	From this study they suggested count distribution algorithm as the best way to parallelize the Apriori algorithm. Hadoop has three mode of operation such as Standalone, Pseudo- Distributed mode, and Fully-Distributed mode. The count distribution strategy was chosen for implementing the Apriori algorithm on a Hadoop cluster.
[1]	"Apriori-Map/Reduce Algorithm"	To illustrate its time complexity which theoretically shows that the algorithm has much higher performance than the sequential algorithm when the map and reduce nodes get added.
[6]	"An efficient implementation of Apriori algorithm based on Hadoop-MapReduce model"	To compare and prove the good performance with proposed 2-phase new algorithm with earlier existing 1-phase and k-phase scanning algorithm repeatedly changing the number of transaction and minimum support.
[5]	"Data Mining Using Cloud: An Experimental Implementation of Apriori over MapReduce"	To deploy the revise Apriori algorithm on Amazon Elastic Cloud Computing (EC2) to evaluate the performance varying number of nodes, and min_sup threshold.
[2]	Map/reduce design and implementation of Apriori algorithm for handling voluminous data-sets	To produces all the subsets that would be generated from the given itemset, these subsets are searched against the data sets and frequency is noted. There are huge data itemsets and their subsets, hence they need to search them simultaneously so that search time reduced. Experiment evaluated by changing Hadoop mode with Fully configured multi-node Hadoop with Different System Configuration (FHDSC) and Fully configured multi-node Hadoop with Similar System Configuration (FHSSC) $N = \frac{FHDSC}{FHSSC}$ $FHDSC = FHSSC = log_e N$ Where N is the number of nodes installed in the cluster

CONCLUSION

MapReduce is good choice for parallel processing of massive data on large cluster of commodity computers. In this study, we studied the parallelization of Apriori algorithm on Hadoop and MapReduce framework. The Hadoop and MapReduce is a good platform for computation of frequent itemsets in Apriori algorithm to analyze massive dataset in various repositories. We reviewed various proposed approaches to parallelize Apriori on Hadoop-MapReduce distributed framework. They are implemented using various techniques such as varying the size up, speedup and scale up to deal with large dataset, changing the Hadoop modes (standalone, pseudo-distributed mode, and fully distributed mode), implemented various platform(Hadoop, EC2, Horton works), they theoretically proven the complexity of Apriori algorithm, and convert dataset into Boolean matrix, and so on. We also reviewed the advantages and challenges that exist in using the Hadoop distributed platform.

ACKNOWLEDGMENT

We wish to thank Universiti Kebangsaan Malaysia (UKM) and Ministry of Higher Education Malaysia for supporting this work by research Grants (ERGS/1/2013/ICT07/UKM/02/3).

REFERENCES

[1] J. Woo, "Apriori-Map/Reduce Algorithm," in *The 2012 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA 2012), Las Vegas*, 2012.

[2] A. K. Koundinya, K. Sharma, K. Kumar, and K. U. Shanbag, "Map/Reduce Deisgn and Implementation of Apriori Alogirthm for handling voluminous data-sets," *arXiv preprint arXiv:1212.4692*, 2012.

[3] H. Yu, J. Wen, H. Wang, and L. Jun, "An improved Apriori algorithm based on the Boolean matrix and Hadoop," *Procedia Engineering*, vol. 15, pp. 1827-1831, 2011.

[4] S. Oruganti, Q. Ding, and N. Tabrizi, "Exploring HADOOP as a Platform for Distributed Association Rule Mining," in *FUTURE COMPUTING 2013, The Fifth International Conference on Future Computational Technologies and Applications*, 2013, pp. 62-67.

[5] J. Li, P. Roy, S. U. Khan, L. Wang, and Y. Bai, "Data mining using clouds: An experimental implementation of apriori over mapreduce," in *12th International Conference on Scalable Computing and Communications (ScalCom)*, 2012.

[6] O. Yahya, O. Hegazy, and E. Ezat, "An Efficient Implementation of Apriori algorithm based on Hadoop-Mapreduce Model," *International Journal of Reviews in Computing*, vol. 12, 2012.

[7] N. Li, L. Zeng, Q. He, and Z. Shi, "Parallel implementation of apriori algorithm based on MapReduce," in *Software Engineering, Artificial Intelligence, Networking and Parallel & Distributed Computing (SNPD), 2012 13th ACIS International Conference on*, 2012, pp. 236-241.

[8] J. Han, M. Kamber, and J. Pei, *Data mining, southeast asia edition: Concepts and techniques*: Morgan kaufmann, 2006.

[9] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," in *ACM SIGMOD Record*, 1993, pp. 207-216.

[10] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *Proc. 20th int. conf. very large data bases, VLDB*, 1994, pp. 487-499.

[11] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, pp. 107-113, 2008.

[12] I. A. VAJK, "Performance Evaluation of Apriori Algorithm on a Hadoop Cluster," 2013.

[13] A. B. Patel, M. Birla, and U. Nair, "Addressing big data problem using Hadoop and Map Reduce," in *Engineering (NUiCONE), 2012 Nirma University International Conference on*, 2012, pp. 1-5.

[14] F. Kovacs and J. Illes, "Frequent itemset mining on hadoop," in *Computational Cybernetics (ICCC), 2013 IEEE 9th International Conference on*, 2013, pp. 241-245.

[15] M. P. Modgi and D. Vaghela, "Mining Distributed Frequent Itemset with Hadoop," 2014.

[16] T. A. S. Foundation. (2014, 14 February). *Welcome to Apache™ Hadoop®!* Available: http://hadoop.apache.org/

[17] A. Holmes, *Hadoop in practice*: Manning Publications Co., 2012.

[18] T. White, *Hadoop: The definitive guide*: " O'Reilly Media, Inc.", 2012.

[19] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "The KDD process for extracting useful knowledge from volumes of data," *Communications of the ACM*, vol. 39, pp. 27-34, 1996.

About Author (s):

Sayeth Saabith received the BSc in 2005 from South Eastern University of Sri Lanka. He is currently an MSc student in Center for Software Technology Management, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia. His research interests are in Cloud computing, Big Data analytics, Data mining.

Elankovan Sundararajan received the BSc and MSc in 1996 from Universiti Kebangsaan Malaysia and the Ph.D. degree in 2008 from The University of Melbourne, Australia. He is currently a Senior Lecturer in the School of Information Technology, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia. He is a founding member of the Distributed and Platform Technology Group within the Software Technology and Management Centre. His research interests are in the parallel and distributed computing, performance of large scale systems and numerical methods. He is a member of IEEE Computer Society since 2005.

Prof Dr. Azuraliza Abu Bakar is a professor at the Center for Artificial Intelligence Technology, Faculty of Information Science and Technology, University Kebangsaan Malaysia. She received her PhD in Artificial Intelligence from University Putra Malaysia. Her research interest are Data Mining, Outbreak Detection Analytics and Climate Change Informatics.