Volume 2, Issue 1, January 2012 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

# Data Mining for High Performance Data Cloud using Association Rule Mining

[1]T.V.Mahendra [2]N.Deepika [3]N.Keasava Rao

[1]Professor & HOD, IT, Narayana Engg. College, Nellore, AP, India
[2] Sr.Assistant Professor, Dept. of ISE, New Horizon College of Engineering, Bangalore, India.
[3]Associate Professor, IT, Narayana Engg. College, Gudur, AP, India.

***ABSTRACT: -*** ***Cloud computing has demonstrated that processing very large datasets over commodity clusters can be done by giving the right programming model. In this paper we have discussed an algorithm to mine the data from the cloud using sector/sphere framework with association rules. Data mining is the process of analyzing data from different perspectives and summarizing it into useful information. Mining association rules is one of the most important aspects in data mining. Association rules are dependency rules which predict occurrence of an item based on occurrences of other items. Apriori is the best-known algorithm to mine association rules. Cloud can be meant as an infrastructure that provides resources and/or service over the internet. A cloud can be a storage cloud that provides block or file based storage service or it can be a compute cloud that provides computational services. Moreover in this paper we have reviewed the design and implementation of sector storage cloud and sphere compute cloud. Sector is the distributed file system, while sphere is the parallel in-storage data processing framework that can be used to process data stored in sector***

***Keywords: Cloud, Apriori, Sphere, Sector, Association Rules***

## 1. INTRODUCTION

High performance data mining system is designed for taking advantage of powerful and shared pools of processors. In that data is distributed over the processors and the computation is done using message passing paradigm. Then all the computation results are gathered and this process is repeated on the new data on the processor. Data mining is the process of analyzing data from different perspectives and summarizing it into useful information - information that can be used to increase revenue, cuts costs, or both. It allows users to analyze data from many different dimensions or angles, categorize it, and summarize the relationships identified. Mining Association rule is a way to find interesting associations among large sets of data items. Using this we have determined the frequent item sets based on a predefined support [6].

By cloud we can say that it is an infrastructure that consists of services delivered through shared datacenters and appearing as a single point of access for consumers' computing needs and also provides demanded resources and/or service over the internet. Sector storage cloud is a distributed storage system that can be deployed over a wide area network and allows users to consume

and download large dataset from any location with a high-speed network connection to the system. Sector automatically replicates files for the better reliability, access and availability. Sphere compute cloud is a computation service which is built on the top of the sector storage cloud. It allows developers to write certain distributed data intensive parallel applications with several simple APIs. Data locality is the key factor for the performance in the Sphere. Thus to summarize we can say that sector manages data in form of distributed indexed files, sphere processes that data using sphere processing engine that is applied parallel on every data segment managed by sector [6].

In this paper we have described the design of storage and compute cloud. We have also describe a data mining application developed using sector and sphere that searches for evolving behavior in distributed network data.

## 2. BACKGROUND AND RELATED WORK

By cloud we can say that it is an infrastructure that consists of services delivered through shared datacenters and appearing as a single point of access for consumers' computing needs and also provides demanded resources and/or service over the internet. A cloud can be a storage cloud that provides block or file based storage service or it can be a compute cloud that provides computational services. It can also be a data cloud that provides record-based, column-based or object-based services. These all types of clouds are set up as a stack of cloud services that provides computing platform to develop cloud based applications.

| | | |
|---|---|---|
| Application 1 | …… | Application N |
| Compute Cloud Services | | |
| Data Cloud Services | | |
| Storage Cloud Services | | |

**Figure 1: A data stack for a cloud consist of layered services**

In this paper we have described a Sector storage cloud and a sphere data cloud. Sector/Sphere is a software platform that supports very large distributed data storage and simplified distributed data processing. The system consists of Sector, a distributed storage system, and Sphere, a runtime middleware to support simplified development of distributed data processing.

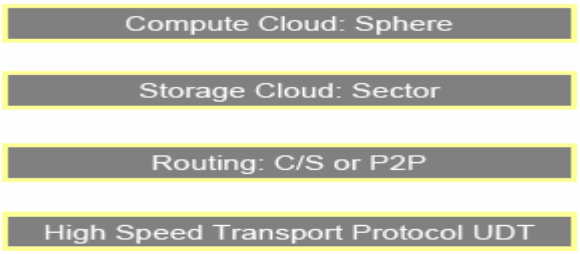


**Figure 2: Sector/Sphere Layered Architecture**

Implicit assumption with most data mining system that have been developed for the computer clusters and grid is that the processors are the scarce resource, and hence it have to be shared. Because of this assumption two types of models generated. One is supercomputing model in which data is moved to the processor whenever the processor became free and available to use. Second is data center model in which we are storing the data and co-locating the computation with the data whenever possible. In supercomputing model, for too many computations, a good portion of time is spent on the transporting data to the available processor. While in the sector and sphere framework, the data is stored persistently and processed in its place whenever possible. So in this data have to wait for the task. The storage clouds provided by Google File System (GFS) [1] and Hadoop Distributed File System (HDFS) [2] support this model. The Compute cloud MapReduce, Data Cloud Hadoop and their underlying storage system GFS and HDFS are specifically designed for computer clusters in data centers. These both systems use information of cluster and computer rack to position file blocks and its replicas. But this doesn't support loosely coupled distributed environments. GFS and HDFS are more designed for tightly coupled systems that are managed by Master node. This is not the case with sector. Sector is designed to support loosely coupled distributed system with peerto- peer architecture [5].

GFS and HDFS storage cloud has assumed that bandwidth of the network connecting different computer clusters containing data is relatively small. But the sector storage cloud is designed for the wide area high performance network which uses specialized UDT protocol. Sector assumes the data is divided in to files while the GFS and HDFS divide the data into blocks. MapReduce computation is most commonly used over GFS and HDFS storage clouds. Using MapReduce first of all Map operation is performed in which all the relevant data is extracted in parallel over multiple nodes. Then shuffling is done in which the data is transported to other nodes as required and finally Reduce operation is performed in which the data is processed over multiple nodes to produce a result set. While in the Sphere compute cloud arbitrary user defined operations is performed to replace both the map and reduce operations. In addition Sector and Sphere both uses the same specialized network transport protocols so that any transfer of data required by Sphere's user defined functions can be transferred efficiently over wide area high performance networks.

## 3. DESIGN OF SPHERE

### 3.1 OVERVIEW

Sphere is a compute cloud that is layered over the Sector storage cloud. Let's take an example to introduce sphere. Assume we have 1 billion images and our goal is to find a specific object in these images. Suppose the image size is 1 MB so that the total data size is 1 TB. The whole image dataset is stored in 64 files named img1.dat, … , img64.dat, each containing one or more images. We can build an index file for each file for accessing an image randomly in the dataset consisting 64 files. The index file indicates the offset and the size of each record in the data file. The index files are named by adding .idx as postfix to the data file name like img1.dat.idx. To use Sphere, the user writes a function "findSpecifiedObject" to find specified object from each image. In this function we will pass image as an input and the output will be our specified object.

findSpecifiedObject (input, output);

The standard serial program will be look like:

For each file F in (imgs slices)
For each image I in F findSpecifiedObject (I, …)
In Sphere client API the corresponding pseudo code looks like:

```
SphereStream imgs;
imgs.init (/*list of IMAGE slices */);
SphereProcess myproc;
myproc.run(imgs, "findSpecifiedObject");
myproc.read (result);
```

In the pseudo code fragment shown above, "imgs" is a Sector stream data structure that stores the metadata of the sector slice files. The Stream is initialized by the application by giving it a list of file names. The Sphere automatically retrieves the metadata from the sector network. The last three line of code specifies the start of the job and wait for the result with the use of sector APIs. User doesn't need to move or locate the data. User also doesn't have to take care about the message passing, scheduling, and fault tolerance. Data locality is a key factor for the performance of Sphere. In addition, sphere provides transparent load balance and fault tolerance [8].

### 3.2 THE COMPUTING PARADIGM

Sphere allows developers to write certain distributed data parallel applications with several simple APIs. The computation paradigm used by sphere is based upon the following concepts. A Sphere database consists of one or more physical files. Computation in sphere is done by User-define function. User-define functions can be independently applied to each element in the dataset, while the result can be written to either the local disk or common destination files on other nodes.

The user defined function is also known as sphere operator that takes a sphere stream as an input and gives sphere stream as an output. Sphere stream is again split into one or two data segments that are processed by sphere servers called Sphere Processing Elements (SPE). The key abstraction used in sphere can be explained as below. A stream is an abstraction in Sphere and it represents dataset or a part of dataset. A Sphere stream consists of multiple data segments and the segments are processed by sphere Processing Engines using slaves.

Figure 3 illustrates how Sphere processes the segments in a stream. Generally there are many more segments than the SPEs but it handles all the segments with load balancing. Slow SPEs simply processes fewer segments. The output segments can in turn be the input to another sphere process. Sphere processes multiple input streams at the same time. After processing SPEs, the output can be sent to multiple locations, rather than just writing on the local disk. This process is known as shuffling. For example, a user defined function specifies an ID for each record in the output and the sphere will send this record to its destination. At the destination, sphere receives results from many SPEs and writes them into file in the same order that they arrive. Sphere also supports extension of this model.

Figure 4 shows that extension. It shows the example that uses two sphere processes. The first sphere process performs hashing in which the input data is hashed into multiple buckets. The hashing function scans whole stream of data and places each element in a proper bucket. If we have all integer data stream then values less then t0 is put in bucket b0 and others are placed in bucket b1. In second sphere process, we are performing sort operation in each bucket. In this stage the process sorts the whole data segment not individual record. SPE is started by a sphere server when a sphere client request for it. Each SPE is based on a user-defined function. The user-defined function that is the Sphere operator is implemented as a dynamic library and is stored on the server's local disk. Uploading such library files on server is limited because of the security reasons[8].

Once the client's request is accepted by the sphere server, it starts an SPE and binds it to the local Sphere operator.
SPE consists of the following four steps:
1. The Client sends a new data segment to the SPE. The data segment contains the filename, offset, number of rows to be processed and additional parameters.
2. The SPE reads the data segment from a remote disk managed by Sector.
3. For each data segment the user defined function processes the data segment and writes the result to a temporary buffer.
4. When the whole data segment is completely processed, the SPE sends an acknowledgement to the client and then writes the results to the appropriate destination

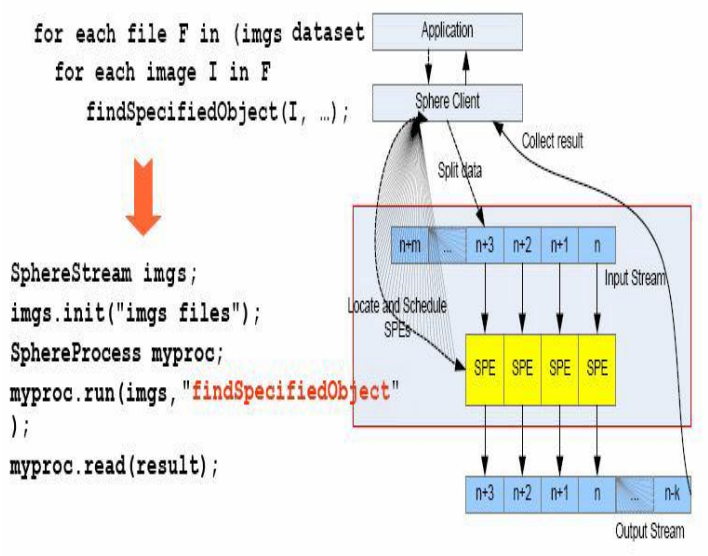


**Figure 3: The Computing paradigm of Sphere**

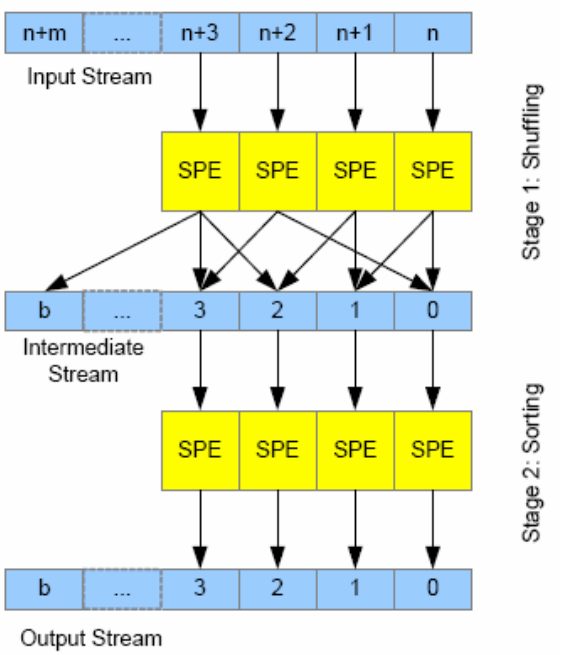


**Figure 4: Sorting large distributed datasets with Sphere**

## 4. DESIGN OF SECTOR

### 4.1 OVERVIEW

Sector is a storage cloud that provides storage service to the cloud. Sector makes some assumption:
1. First assumption of Sector is that it has access to a large number of commodity computers.
2. Second assumption of sector is that various nodes in the system are connected through a high speed network.
3. Third assumption of sector is that dataset it stores are divided into 1 or more files.

Figure 5 shows the overall architecture of the sector. As shown in the figure, the architecture contains mainly security server, Master server, Slaves and client. The security server maintains all user accounts, their passwords and file access information. It also maintains list of IP addresses of the authorized slave nodes, so no unauthorized computer can join the system. The master server maintains the metadata of the files stored in the system which controls the running of all slave nodes. The master node communicates with the server node to verify the slaves, the client and the users. The slaves are the nodes which are connected to master node. It stores all the files manage by the system. Generally the slaves are running on computer cluster that are located in one or more data centers [8].

To summarize, the Sector system contains, a master node that maintains the file system, while the data is stored on the slave nodes, possibly across multiple data centers. A security server provides user account verification, access control IP list, etc. UDT [3] is used for high speed data transfer between slaves and between slaves and clients. The Sector is not a native file system but it provides services that rely in part on the local native file systems. In Sector, only the users in the community who have been added to the Sector access control only can write into the Sector. Any members that are not in the community or from the public can read data, unless additional restrictions are imposed [7].

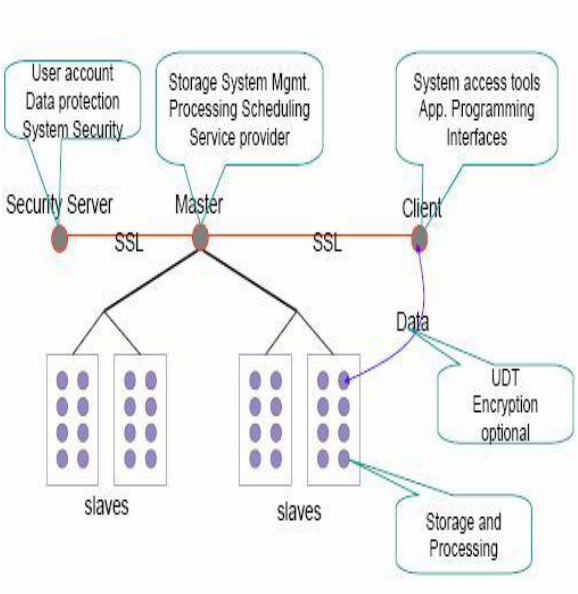


**Figure 5: The Sector System Architecture**

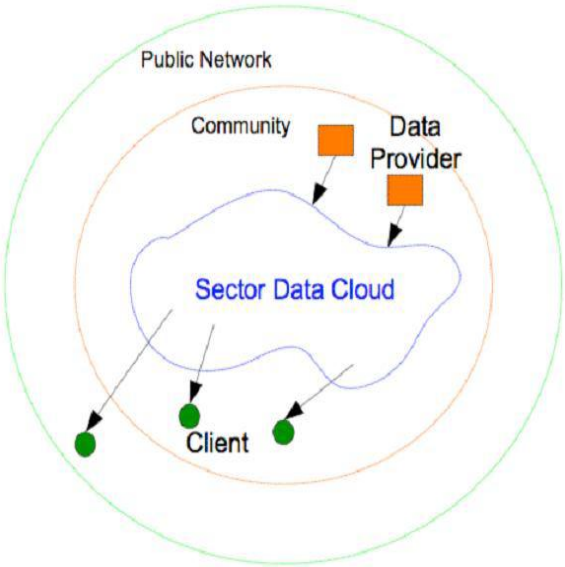


**Figure 6: Sector Data Cloud Accessibility to the user.**

A sector can be access by the Sector client as follows:
1. The Sector client connects to a known Sector server Ss, and requests the locations of an entity managed by the Sector using the entity's name.
2. The Sector Server Ss runs a look-up inside the server network using the services from the routing layer and returns one or more locations to the client. In general, an entity managed by Sector is replicated several times within the Sector network. The routing layer can use information involving network bandwidth

and latency to determine which replica location should be provided to the client.
3. The client requests a data connection to one or more servers on the returned locations using a specialized Sector library designed to provide efficient message passing between geographically distributed nodes. The Sector library used for messaging uses a specialized protocol developed for Sector called the Group Messaging Protocol.
4. All further requests and responses are performed using a specialized library for high performance network transport called UDT [3]. UDT is used over the data connection established by the message passing library.

## 5. ASSOCIATION RULES

### 5.1 OVERVIEW

Association rule is very popular and well researched method for discovering interesting relations between variables in large databases. Given a set of transactions, where each transaction is a set of items, an association rule is an expression X => Y, where X and Y are sets of items. The intuitive meaning of such a rule is that transactions in the database which contain the items in X tend to also contain the items in Y. An example of such a rule might be that 98% of customers that purchase tires and auto accessories also buy some automotive services; here 98% is called the confidence of the rule. The support of the rule X => Y is the percentage of transactions that contain both X and Y.
The problem of mining association rules is to find all rules that satisfy a user-specified minimum support and minimum confidence. Applications include cross marketing, attached mailing, catalog design, loss-leader analysis, add-on sales, store layout, and customer segmentation based on buying patterns [9].

The problem of mining association rules can be decomposed into two sub problems:
1. Find all sets of items (item sets) whose support is greater than the user-specified minimum support. Item sets with minimum support are called frequent
item sets.
2. Use the frequent item sets to generate the desired rules. The general idea is that if, say, ABCD and AB are frequent item sets, then we can determine if the rule AB => CD holds by computing the ratio conf = support(ABCD)/support(AB). If conf >= minimum confidence, then the rule holds. That is the rule will have minimum support because ABCD is frequent. Much of the research has been focused on the first sub problem as the database is accessed in this part of the computation and several algorithms have been proposed. In association rule mining algorithm, the most algorithms are based on Apriori algorithm to calculate, and in the mining process they can produce amount of option set which reduce the efficiency of association rule mining.

### 5.2 APRIORI ALGORITHM

Apriori is designed to operate on databases containing transactions. Apriori uses a "bottom up" approach, where frequent subsets are extended one item at a time and groups of candidates are tested against the data. The algorithm terminates when no further successful extensions are found. Apriori uses breadth-first-search and a tree structure to count candidate item sets efficiently. It generates candidate item sets of length $k$ from item sets of length $k-1$. Then it prunes the candidate which has an infrequent sub pattern. According to the downward closure lemma, the candidate set contains all frequent $k$-length item sets. After that, it scans the transaction database to determine frequent item sets among the candidates. The above code gives an overview of the Apriori algorithm. The first pass of the algorithm simply counts item occurrences to determine the frequent itemsets.

A subsequent pass, say pass k, consists of two phases. First, the frequent itemsets Lk-1 found in the (k-1)th pass are used to generate the candidate itemsets Ck, using the Apriori candidate generation procedure. Next, the database is scanned and the support of candidates in Ck is counted. For fast counting, we need to efficiently determine the candidates in Ck contained in a given transaction t. A hash-tree data structure is used for this purpose.

$L_1 := \{\text{frequent 1-itemsets}\};$
$k := 2;$ // k represents the pass number
**while** $( L_{k-1} \neq \emptyset )$ **do**
**begin**
  $C_k :=$ New candidates of size $k$ generated from $L_{k-1}$;
  **forall** transactions $t \in \mathcal{D}$ **do**
    Increment the count of all candidates in $C_k$ that are contained in $t$;
  $L_k :=$ All candidates in $C_k$ with minimum support;
  $k := k + 1;$
**end**
Answer $:= \bigcup_k L_k;$

## 6. PROPOSED SYSTEM

We are going to develop the data mining on the high performance data cloud using Sector/Sphere framework and Association rule. Below are the steps to develop it.
1. Select the minimum support threshold (Ts) and minimum confidence threshold (Tc), minimum data size (Sizemin) and maximum data size (Sizemax).
2. We now input the data stream to the sphere processing elements. The stream is divided into data segments. The number of data segments per SPE is calculated on the basis of number of SPE and the entire stream size. Data segments from the same file are not processed at the same time until other SPE become idle.
3. The SPE accepts a new data segment from the client, which contains the file name, offset, number of rows to be processed, and additional parameters.
4. The SPE reads the data segment and its record index from local disk or from a remote disk managed by sector.

5. For each data segment find out the frequent term set with length of 1 which is recorded as L1 , L1 is used to find the aggregate L2 of frequent 2-term sets, L2 is used to find the aggregate L3 of frequent 3-term sets, and so the cycle continues, until no new frequent k - term sets can be found.
6. We generate strong association rules on the basis of the found frequent term sets i.e. we generate those association rules whose support and confidence respectively greater than or equal to the pre-given support threshold (Ts) and confidence threshold (Tc).
7. For each data segment (single data record, group of data records, or entire data file), the Sphere operator processes the data segment using the association rules and writes the result to a temporary buffer. In addition, the SPE periodically sends acknowledgments and feedback to the client about the progress of the processing.
8. When the data segment is completely processed, the SPE sends an acknowledgment to the client and writes the results to the appropriate destinations, as specified in the output stream. If there are no more data segments to be processed, the client closes the connection to the SPE, and the SPE is released.

**7. CONCLUSION**

For several years now, the commodity clusters have been quite common and over the next several years, wide area high performance networks will begin to connect these clusters. We are moving towards the era in which there are large distributed datasets that must be persisted on the disk for large amount of time and we need a high performance computing paradigm that moves data as small as possible. By reviewing Sector and Sphere we come to know that Sector and Sphere are designed for these types of applications. In this paper we have also discussed about the integration of Sector/Sphere framework and Association rule. This enables the application of association rule algorithm to the wide range of cloud services available on the web.

**8. REFERENCES**

[1] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. "The Google File System". In *SOSP*, 2003.
[2] Dhruba Borthaku. "The hadoop distributed file system: Architecture and design". retrieved fromlucene.apache.org/hadoop, 2007.
[3] Yunhong Gu and Robert L. Grossman. "UDT: UDPbased data transfer for high-speed wide area networks". *Computer Networks*, 51(7):1777—1799, 2007.
[4] Han J , Kamber M. "Data Mining: Concepts and Techniques". 2/e San Francisco: CA Morgan Kaufmann Publishers, an imprint of Elsevier. pp- 259-261, 628-640 (2006)
[5] Robert L. Grossman and Yunhong Gu "Data Mining using high performance data clouds: Experimental Studies using Sector and Sphere" Retrieved from http://sector.sourceforge.net/pub/grossman-gu-ncdm-tr-08-04.pdf.
[6]Kanhaiya lal and N. C. Mahanti, "A Novel Data Mining Algorithm for Semantic Web Based Data Cloud".
[7]The Sector Project. Sector, a distributed storage and computing infrastructure, version 1.4.
[8] Yunhong Gu and Robert L. Grossman "Sector and Sphere: The Design and Implementation of a High Performance Data Cloud".
[9] Ramakrishnan Srikant and Rakesh Agrawal "Mining Generalized Association Rules".

## About Authors

**1 T.V.MAHENDRA** Working as Professor & Head of Information Technology in Narayana Engineering College, Nellore . He completed his M.Tech from Nagarjuna University in 2006 in Computer Science and Engineering. He has richest experience in teaching of 12 years. He guided 10 projects under UG/PG level. His area of interest is "Data Mining"(Association Rule Mining). He acted as reviewer for industrial oriented mini projects. He presented papers in 3 national and international conferences.

**2.N.Deepika**, Working as Sr.Asst.Professor in the department of ISE at New Horizon College of Engineering, Bangalore, India. Her research interests include Data Mining and Mobile Networks.

**3. N.Kesava Rao**, received his M.E in CSE from Sathyabama University, Chennai. He is working as Associate Professor in IT at Narayana Engineering College, Gudur, AP, India. He has published few papers in the area of data mining.