A Novel Data Mining Algorithm for Semantic Web Based Data Cloud

Kanhaiya Lal klal@bitmesra.ac.in
Sr. Lecturer/Dept. of Computer Sc. & Engg.,
Birla Institute of Technology, Patna Campus,
Patna, 800014, India

N.C.Mahanti ncmahanti@rediffmail.com
Professor & Head/Department of Applied Mathematics
Birla Institute of Technology, Mesra
Ranchi, 835215, India

Abstract

By a cloud, we mean an infrastructure that provides resources and/or services over the Internet. A storage cloud provides storage services, while a compute cloud provides compute services. We describe the design of the Sector storage cloud and how it provides the storage services required by the Sphere compute cloud [14]. Different efforts have been made to address the problem of data mining in the cloud framework. In this paper we propose an algorithm to mine the data from the cloud using sector/sphere framework and association rules. We also describe the programming paradigm supported by the Sphere compute cloud and Association rules. Sector and Sphere are discussed for analyzing large data sets using computer clusters connected with wide area high performance networks

Keywords: Cloud, Web, Apriori, Sphere, Association, Sector, confidence, support.

1. INTRODUCTION

Data mining is a treatment process to extract useful and interesting knowledge from large amount of data. The knowledge modes data mining discovered have a variety of different types. The common patterns are: association mode, classification model, class model, sequence pattern and so on.
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. It is simple but effective and can help the commercial decision making like the storage layout, appending sale and etc. We usually use distributed system as a solution to mining association rules when mass data is being collected and warehoused. With the development of web and distributed techniques, we begin to store databases in distributed systems. Thus researches on the algorithm of mining association rules in distributed system are becoming more important and have a broad application foreground. Distributed algorithm has characters of high adaptability, high flexibility, low wearing performance and easy to be connected etc. [15].

The Semantic Web is based on a vision of Tim Berners-Lee, the inventor of the WWW. The great success of the current WWW leads to a new challenge: A huge amount of data is interpretable by humans only; machine support is limited. Berners-Lee suggests enriching the Web by machine-process able information which supports the user in his tasks. For instance, today's search engines are already quite powerful, but still too often return excessively large or inadequate lists of hits. Machine process able information can point the search engine to the relevant pages and can thus improve both precision and recall. For instance, today it is almost impossible to retrieve information with a keyword search when the information is spread Fig. 1. The layers of the Semantic Web over several pages. Consider, e.g., the query for Web Mining experts in a company intranet, where the only explicit information stored are the relationships between people and the courses they attended on one hand, and between courses and the topics they cover on the other hand. In that case, the use of a rule stating that people who attended a course which was about a certain topic have knowledge about that topic might improve the results. The process of building the Semantic Web is currently an area of high activity. Its structure has to be defined, and this structure then has to be filled with life. In order to make this task feasible, one should start with the simpler tasks first. The following steps show the direction where the Semantic Web is heading:

1. Providing a common syntax for machine understandable statements.
2. Establishing common vocabularies.
3. Agreeing on a logical language.
4. Using the language for exchanging proofs.

Berners-Lee suggested a layer structure for the Semantic Web. This structure reflects the steps listed above. It follows the understanding that each step alone will already provide added value, so that the Semantic Web can be realized in an incremental fashion.[17]

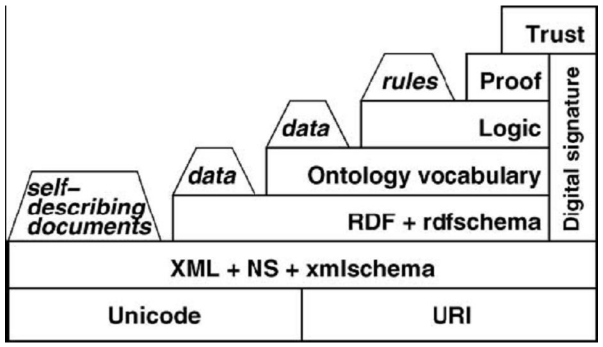

Figure 1. The layers of the Semantic Web.

The paper reveals a distributed high performance data mining system called Sector/Sphere that is based on an entirely different paradigm. Sector is designed to provide long term persistent storage to large datasets that are managed as distributed indexed files. In this paper, it has been described the design of Sector/Sphere. We also describe a data mining application developed using Sector/Sphere that searches for emergent behavior in distributed network data. Different segments of the file are scattered throughout the distributed storage managed by Sector. Sector generally replicates the data to ensure its longevity, to decrease the latency when retrieving it. and to provide opportunities for parallelism. Sector is designed to take advantage of wide area high performance networks when available [14]. The data is persistently stored and processed in place whenever possible. In this model, the data waits for the task or query. The storage clouds provided by Amazon's S3 [1], the Google File System [2], and the open source Hadoop Distributed File System (HDFS) [3] support this model. With the Sector/Sphere software from Source Forge, Terasort and Terasplit benchmarks, and the Angle datasets from the Large Data Archive, the algorithm may be implemented.

2. BACKGROUND & RELATED WORK

Cloud means, an infrastructure that provides resources and/or services over the Internet. A *storage cloud* provides storage services (block or file based services); a *data cloud* provides data management services (record-based, column-based or object-based services); and a *compute cloud* provides computational services. Often these are layered (compute services over data services over storage service) to create a stack of cloud services that serves as a computing platform for developing cloud-based applications [14].

Examples include Google's Google File System (GFS), BigTable and MapReduce infrastructure [4]; Amazon's S3 storage cloud, SimpleDB data cloud, and EC2 compute cloud [5]; and the open source Hadoop system [3]. In this section, we describe some related work in high performance and distributed data mining. For a recent survey of high performance and distributed data mining systems, see [6].

By and large, data mining systems that have been developed for clusters, distributed clusters and grids have assumed that the processors are the scarce resource, and hence shared. When processors become available, the data is moved to the processors, the computation is started, and results are computed and returned [7]. In practice with this approach, for many computations, a good portion of the time is spent transporting the data.

Key Characteristics

Agility: Agility improves with users' ability to rapidly and inexpensively re-provision technological infrastructure resources.

Cost: Cost is claimed to be greatly reduced and capital expenditure is converted to operational expenditure. This ostensibly lowers barriers to entry, as infrastructure is typically provided by a third-party and does not need to be purchased for one-time or infrequent intensive computing tasks. Pricing on a utility computing basis is fine-grained with usage-based options and fewer IT skills ,are required for implementation (in-house).

Device and location independence enable users to access systems using a web browser regardless of their location or what device they are using (e.g., PC, mobile). As infrastructure is off-site (typically provided by a third-party) and accessed via the Internet, users can connect from anywhere.

Multi-tenancy enables sharing of resources and costs across a large pool of users. One of the most compelling reasons for vendors/ISVs to utilize multi-tenancy is for the inherent data aggregation benefits. Instead of collecting data from multiple data sources, with potentially different database schemas, all data for all customers is stored in a single database schema. Thus, running queries across customers, mining data, and looking for trends is much simpler. This reason is probably overhyped as one of the core multi-tenancy requirements is the need to prevent Service Provider access to customer (tenant) information.

Centralization of infrastructure in locations with lower costs (such as real estate, electricity, etc.)

Peak-load capacity increases highest possible load-levels.

Utilization and efficiency improvements for systems that are often only 10–20% utilized.

Reliability improves through the use of multiple redundant sites, which makes cloud computing suitable for business continuity and disaster recovery. Nonetheless, many major cloud computing services have suffered outages, and IT and business managers can at times do little when they are affected.

Scalability via dynamic ("on-demand") provisioning of resources on a fine-grained, self-service basis near real-time, without users having to engineer for peak loads. Performance is monitored and consistent and loosely-coupled architectures are constructed using web services as the system interface.

Security could improve due to centralization of data, increased security-focused resources, etc., but concerns can persist about loss of control over certain sensitive data, and the lack of security for stored kernels. Security is often as good as or better than under traditional systems, in part because providers are able to devote resources to solving security issues that many customers cannot afford. Providers typically log accesses, but accessing the audit logs themselves can be difficult or impossible. Furthermore, the complexity of security is greatly increased when data is distributed over a wider area and / or number of devices.

Sustainability comes through improved resource utilization, more efficient systems, and carbon neutrality. Nonetheless, computers and associated infrastructure are major consumers of energy.

Maintenance cloud computing applications are easier to maintain, since they don't have to be installed on each user's computer. They are easier to support and to improve since the changes reach the clients instantly.

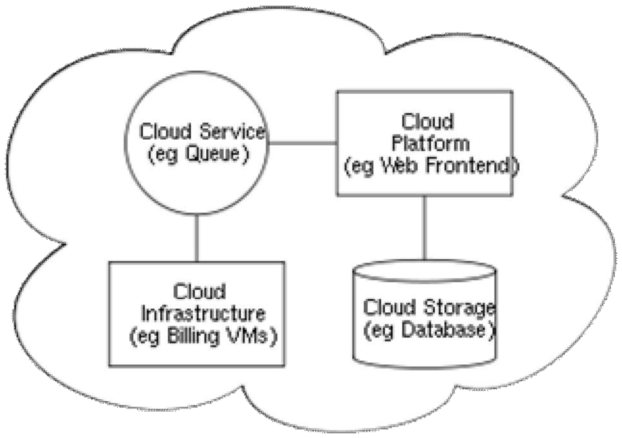
Cloud Service (eg Queue)
Cloud Platform (eg Web Frontend)
Cloud Infrastructure (eg Billing VMs)
Cloud Storage (eg Database)

FIGURE 2: Cloud Computing Model

Cloud Computing Models

Cloud computing is a highly touted recent phenomenon. As noted, there is little hope of obtaining consensus or a standard definition regarding exactly what constitutes a ‘‘cloud” (and the term ‘‘grid” has been similarly overloaded). For example, emphasizes quality of service contracts for a cloud, contrasts social issues with technical infrastructure, while others focus on price or on the nature of the resources provided (e.g., storage, processors, platforms, or application services). Some writers emphasize what the cloud provides to its consumers, e.g., services on demand. Others emphasize what is underneath—a warehouse full of servers. The following features, especially the first three, are commonly associated with clouds. A consumer can be an individual lab, a consortium participant, or a consortium. _ Resource outsourcing: Instead of a consumer providing their own hardware, the cloud vendor assumes responsibility for hardware acquisition and maintenance. _ Utility computing: The consumer requests additional resources as needed, and similarly releases these resources when they are not needed. Different clouds offer different sorts of resources, e.g., processing, storage, management software, or application services . Large numbers of machines: Clouds are typically constructed using large numbers of inexpensive machines.
As a result, the cloud vendor can more easily add capacity and can more rapidly replace machines that fail, compared with having machines in multiple laboratories. Generally speaking

these machines are as homogeneous as possible both in terms of configuration and location. _ Automated resource management: This feature encompasses a variety of configuration tasks typically handled by a system administrator. For example, many clouds offer the option of automated backup and archival. The cloud may move data or computation to improve responsiveness. Some clouds monitor their offerings for malicious activity. _ Virtualization: Hardware resources in clouds are usually virtual; they are shared by multiple users to improve efficiency. That is, several lightly-utilized logical resources can be supported by the same physical resource. _ Parallel computing: Map/Reduce and Hadoop are frameworks for expressing and executing easily-parallelizable computations, which may use hundreds or thousands of processors in a cloud.[16]

To enable a holistic enterprise-modeling of software assets and facilitate the generalization of services across an organization and even beyond its boundaries, the Service-oriented_modeling framework (SOMF) offers virtualization capabilities.

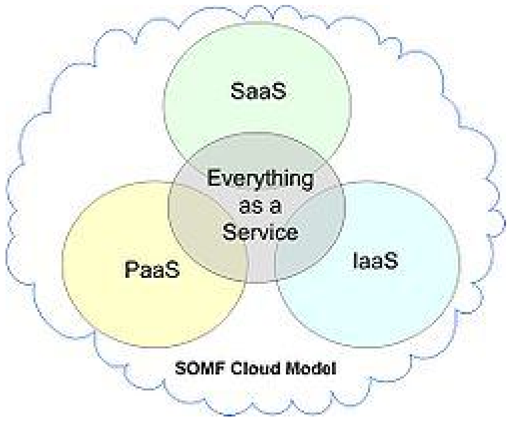

FIGURE 3: SOMF Cloud Computing Model

This ability to abstract services in spite of their location, interoperability challenges, or contribution to an architecture model fosters an elastic Cloud Computing Environment (CCE) that is nimble enough to adapt to changes and vital to business or technological imperatives.

Moreover, the Service-oriented modeling framework (SOMF) as an enterprise modeling language generalizes services. Thus, the notion of “Everything-as-a-Service” encompasses the cloud computing distributed entity model (as illustrated on the far right): infrastructure-as-a-Service (IaaS), platform-as-a-service (PaaS), and software-as-a-service (SaaS). These can be modeled by SOMF as it conceives a service as an abstracted organizational asset and not necessarily as a Web service.

Modeling a cloud computing not only requires a language that must be able to abstract services and an environment that is typically virtual, but also hide the implementation from consumers. SOMF offers these abstraction capabilities by elevating the abstraction level of an organizational asset to enable higher cloud computing reusability rates.

Types by visibility:

Public cloud

Public cloud or *external cloud* describes cloud computing in the traditional mainstream sense, whereby resources are dynamically provisioned on a fine-grained, self-service basis over the Internet, via web applications/web services, from an off-site third-party provider who shares resources and bills on a fine-grained utility computing basis

Hybrid cloud

A hybrid cloud environment consisting of multiple internal and/or external providers will be typical for most enterprises. A hybrid cloud can describe configuration combining a local device, such as a Plug computer with cloud services. It can also describe configurations combining virtual and physical, collocated assets—for example, a mostly virtualized environment that requires physical servers, routers, or other hardware such as a network appliance acting as a firewall or spam filter

Private cloud

Private cloud and internal cloud are neologisms that some vendors have recently used to describe offerings that emulate cloud computing on private networks. These products claim to deliver some benefits of cloud computing without the pitfalls, capitalising on data security, corporate governance, and reliability concerns. They have been criticized on the basis that users "still have to buy, build, and manage them" and as such do not benefit from lower up-front capital costs and less hands-on management, essentially lacking the economic model that makes cloud computing such an intriguing concept.

While an analyst predicted in 2008 that private cloud networks would be the future of corporate IT, there is some uncertainty whether they are a reality even within the same firm. Analysts also claim that within five years a "huge percentage" of small and medium enterprises will get most of their computing resources from external cloud computing providers as they "will not have economies of scale to make it worth staying in the IT business" or be able to afford private clouds. Analysts have reported on Platform's view that private clouds are a stepping stone to external clouds, particularly for the financial services, and that future data centres will look like internal clouds.

The term has also been used in the logical rather than physical sense, for example in reference to platform as service offerings, though such offerings including Microsoft's Azure Services Platform are not available for on-premises deployment.

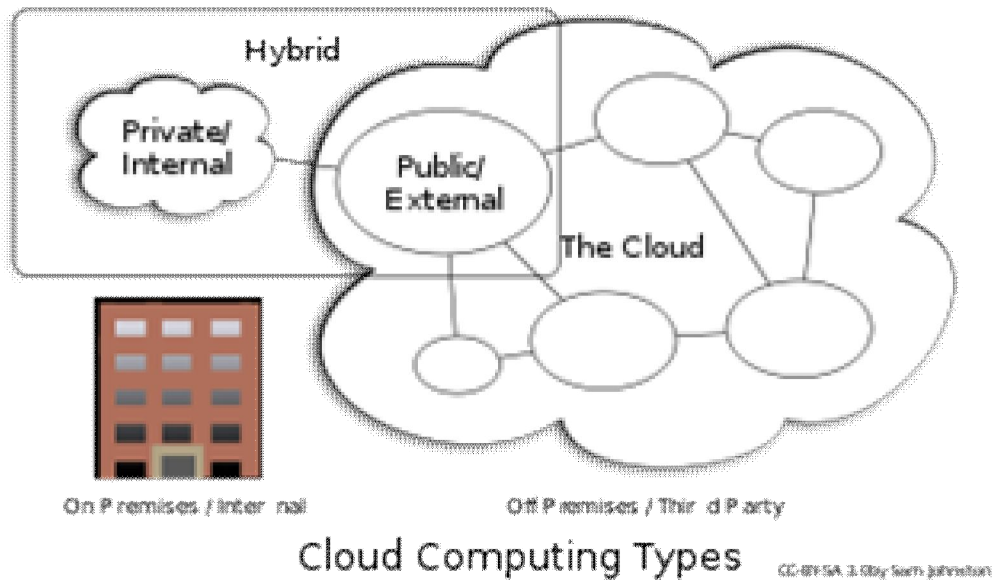

FIGURE 4: Cloud Computing Types

3. ASSOCIATION RULES.

Definition 1 confidence

Set up I={i_1,i_2,i_m}for items of collection, for item in i_j(1≤j≤m), (1≤j≤m) for lasting item, D={T_1,T_N} it is a trade collection, Ti$\subseteq$I (1≤i≤N) here T is the trade. Rule r $\rightarrow$ q is probability that concentrates on including in the trade.

The association rule here is an implication of the form r $\rightarrow$ q where X is the conjunction of conditions, and Y is the type of classification. The rule r $\rightarrow$q has to satisfy specified minimum support and minimum confidence measure .The support of Rule r $\rightarrow$ q is the measure of frequency both r and q in D S(r) = |r|/|D|

The confidence measure of Rule r $\rightarrow$ q is for the premise that includes r in the bargain descend, in the meantime includes q C(r $\rightarrow$ q) =S(r_q) /S(r)

Definition 2 Weighting support

Designated ones project to collect I = {i_1, i_2, i_m}, each project i_j is composed with the value w_j of right(0≤ j ≤1, 1≤j ≤m). If the rule is r $\rightarrow$ q, the weighting support is

$$S_w(r) = \frac{1}{k}\sum_{i\in r} w_j S(r)$$

And, the K is the size of the Set r_q of the project. When the right value w_j is the same as i_j, we are calculating the weighting including rule to have the same support.

Association rule mining is the current hot. In association rule mining algorithms, 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 the association rule mining; at the same time the association rule mining will obtain amount of redundant rules, which will reduce the validity of the association rule mining; and also the user interaction performance of the association rule mining is relatively poor. On the basis of in-depth study on the existing data mining algorithms, according to the disadvantages of the association rule mining algorithms in the relational databases, a new data mining algorithm based on association rule is presented. Existing mining algorithm of association rules can be broadly divided into search algorithms, hierarchical algorithms, data sets partitioning algorithm, and so on[11] .

1) Search algorithms

Search algorithm is to deal with all the term sets contained in the affairs which were read into the data set, so it needs to calculate the support of all term sets in the data aggregate *D*. Search algorithm can find all the frequent term sets only through one scan on data sets, an affair contained *n* projects will generate 2*n* −1 term sets, and when the terms the data set *D* contained is very large, the quantity of the option sets should be calculated and stored is often very large. Therefore, such algorithms can only be applied in the association rule mining with relatively concentrative data.

2) Hierarchy algorithm

The hierarchy algorithm with Apriori algorithm to be representation is to find frequent term sets since childhood and until now in the terms contained. Apriori algorithm will find all the frequent *k* term sets in the first *k* scan on the data sets, the option sets in the first *k* +1 scan will be generated by all frequent *k* term sets through connecting computing. The number Apriori algorithm need to scan data sets is equal to the term numbers of the maximum frequent term sets. The hierarchical algorithm with Apriori algorithm to be representation can generate a relatively small option set, and the number of scanning database is decided by the term numbers of the maximum frequent term sets.

3) Partitioning algorithm

Data set partitioning algorithm includes partition algorithm, DIC algorithm will divide the entire data set into data blocks which can be stored in memory to handle, in order to save the I/O spending of visiting rendering. Partition algorithm only requires two times of scan on the entire data set. DIC algorithm can identify all the frequent term sets through the two times of scan when appropriately dividing data blocks. The number of the option term set of the data set dividing the algorithm generally larger than it of Apriori algorithm, increasing the data distortion can reduce the number of the option term set. Data set partitioning algorithm is the basis of the various parallel association rule mining algorithm and distributed association rule mining algorithm.

4) Apriori Algorithm

Apriori algorithm is an effective algorithm to mine Boolean association rule frequent term sets. Apriori algorithm uses the strategy of breadth-first search, that is, layer-by-layer search iterative method, first of all, find out the frequent term set with length of 1 which is recorded as L_1 , L_1 is used to find the aggregate L_2 of frequent 2-term sets, L_2 is used to find the aggregate L_3 of frequent 3-term sets, and so the cycle continues, until no new frequent *k* - term sets can be found. Finding each L_k needs a database scan. Finding all the frequent term sets is the core of association rule mining algorithm and it has the maximum calculating workload. Afterward, according to the minimum confidence threshold the effective association rules can be constructed from the frequent term sets[10].

5) Sphere

The Sphere Compute Cloud is designed to be used with the Sector Storage Cloud. Sphere is designed so that certain specialized, but commonly occurring, distributed computing operations can be done very simply. Specifically, if a user defines a function ***p*** on a distributed data set ***a*** managed by Sector, then invoking the command

```
sphere.run(a, p);
```

applies the user defined function *p* to each data record in the dataset *a*. In other words, if the dataset *a* contains 100, 000, 000 records *a*[*i*], then the Sphere command above replaces all the code required to read and write the array *a*[*i*] from disk, as well as the loop:

```
for (int i = 0, i < 100000000; ++i)
p(a[i]);
```

The Sphere programming model is a simple example of what is commonly called a stream programming model. Although this model has been used for some time, it has recently received renewed attention due to its use by the general purpose GPU (Graphics Processing Units) community. Large data sets processed by Sphere are assumed to be broken up into several files. For example, the Sloan Digital Sky Survey dataset [12] is divided up into 64 separate files, each about 15.6 GB in size. The files are named sdss1.dat, . . ., sdss64.dat. Assume that the user has a written a function called *find-BrownDwarf* that given a record in the SDSS dataset, extracts candidate Brown Dwarfs. Then to find brown dwarfs in the Sloan dataset, one uses the following Sphere code:

```
 Stream sdss;
sdss.init(...); //init with 64 sdss files
Process* myproc = Sector::createJob();
myproc->run(sdss, "findBrownDwarf");
myproc->read(result);
```

With this code, Sphere uses Sector to access the required SDSS files, uses an index to extract the relevant records, and for each record invokes the user defined function *find- BrownDwarf*. Parallelism is achieved in two ways. First, the individual files can be processed in parallel. Second, Sector is typically configured to create replicas of files for archival purposes. These replicas can also be processed in parallel. An important advantage provided by a system such as Sphere is that often data can be processed in place, without moving it. In contrast, a grid system generally transfers the data to the processes prior to processing [7].

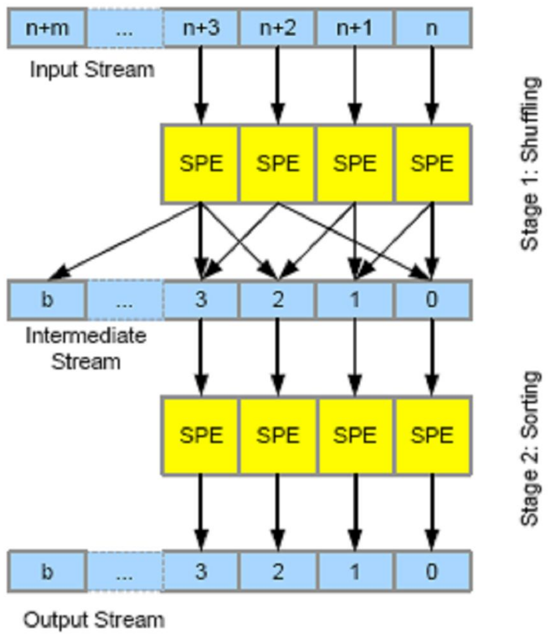

FIGURE 5: Sphere operators process Sphere streams over distributed Sphere Processing Elements (SPE) [14].

6) Sector

Sector storage cloud described as designed for wide area, high performance 10 Gb/s networks and employs specialized protocols, such as UDT to utilize the available bandwidth on these networks. Sector provides services that rely in part on the local native file systems.

The characteristics of sector are

1. Sector is designed to support a community of users, not all of whom may have write access to the Sector infrastructure.

2. Sector provides long term archival storage and access for large distributed datasets.

3. Sector is designed to utilize the bandwidth available on wide area high performance networks.

4. Sector supports a variety of different routing and network protocols. [14].

Sector has separate layers for routing and transport and interfaces with these layers through well defined APIs. In this way, it is relatively straightforward to use other routing or network protocols. In addition, UDT is designed in such a way that a variety of different network protocols can be used simply by linking in one of several different libraries. As an example, Sector is used to archive and to distribute the Sloan Digital Sky Survey (SDSS) to astronomers around the world. Using Sector, the SDSS BESTDR5 catalog, which is about 1.3TB when compressed, can be transported at approximately 8.1 Gb/s over a 10 Gb/s wide area network with only 6 commodity servers [14].

Sector assumes that large datasets are divided into multiple files, say file01.dat, file02.dat, etc. It also assumes that each file is organized into records. In order to randomly access a record in the data set, each data file in Sector has a companion index file, with a post-fix of ".idx". Continuing the example above, there would be index files file01.dat.idx, file02.dat.idx, etc. The data file and index file are always co-located on the same node. Whenever Sector replicates the data file, the index file is also replicated. The index contains the start and end positions (i.e., the offset and size) of each record in the data file. For those data files without an index, Sphere can only process them at the file level, and the user must write a function that parses the file and extracts the data

A Sector client accesses data using Sector as follows:

1. The Sector client connects to a known Sector server S, and requests the locations of an entity managed by Sector using the entity's name.

2. The Sector Server S 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 [8]. UDT is used over the data connection established by the message passing library.

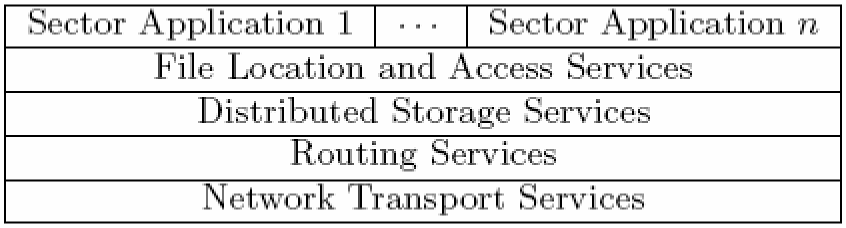
Sector Application 1
···
Sector Application n
File Location and Access Services
Distributed Storage Services
Routing Services
Network Transport Services

FIGURE 6: Sector consist of several layered services

Sector is designed to support a variety of different routing and networking protocols. The version used for the experiments described below are designed to support large distributed datasets, with loose management provided by geographically distributed clusters connected by a high performance wide area network. With this configuration, a peer-to-peer routing protocol (the Chord protocol described in [9]) is used so that nodes can be easily added and removed from the system. The next version of Sector will support specialized routing protocols designed for wide area clouds with uniform bandwidth and approximately equal RTT between clusters, as well as non-uniform clouds in which bandwidth and RTT may vary widely between different clusters of the cloud. Data transport within Sector is done using specialized network protocols. In particular, data channels within Sector use high performance network transport protocols, such as UDT [13]. UDT is a rate-based application layer network transport protocol that supports large data flows over wide area high performance networks. UDT is *fair* to several large data flows in the sense that it shares bandwidth equally between them. UDT is also *friendly* to TCP flows in the sense that it backs off when congestion occurs, enabling any TCP flows sharing the network to use the bandwidth they require. Message passing with Sector is done using a specialized network transport protocol developed for this purpose called the Group Messaging Protocol or GMP. [14].

4. METHODOLOGY

Steps for data mining using sector sphere and association rules

1. Select the minimum support threshold(T_s) and minimum confidence threshold(T_c), minimum data size($Size_{min}$) and maximum data size ($Size_{max}$).
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 L_1 , L_1 is used to find the aggregate L_2 of frequent 2-term sets, L_2 is used to find the

aggregate L_3 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 (T_s) and confidence threshold (T_c).
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.

Pseudocode for data mining using sector sphere & association rules

```
procedure data_mining_cloud()
{
Set minimum data segment size $D_{min}$
Set maximum data segment size $D_{max}$
Set minimum support threshold $T_s$
Set maximum confidence threshold $T_c$
Size of stream=S
Number of SPE=N
$N_d$= S/N;
        for(i=0;i<N;i++)
        {
        $N_{di'}$ =0;
        }

spawn(SPE);
for each SPE while there are data segments to be processed
        {
        start a SPE connection
        If($N_{di'}$ > $N_{di}$)
        return;
        Accept new data segment in Data[$D_{max}$)
        metadata[8]=substr(Data, $D_{max}$-8, $D_{max}$);
        findFrequentTermSets(Data, $D_{max}$ -8);
        sendAcknowledgement($N_{di'}$);
        $N_{di'}$ = $N_{di'}$ +1;
        release SPE connection
        return;
        }

}


procedure findFrequentTermSets(Data, $D_{max}$ -8) // **method 1//**
{
        For(k=2;$L_{k-1}$≠| |); k++)
        {
        $C_k$=apriori_gen($L_{k-1}$);
                for each element t€ Data
                {
                $C_t$=subset($C_k$,t);
                        for each candidate c€ Ct
                        c.count++;
                        end for;

                }
        $L_k$={c € $C_k$ | c.count>$T_s$}
      }
        return L=$U_k$ $L_{k;}$
}



procedure apriori_gen($L_{k-1}$){
 for each itemset l1 €$L_{k-1}$
        for each itemset l2 €$L_{k-1}$
 if(l1[1]=l2[1])^ (l1[2]=l2[2]) ])^ (l1[3]=l2[3]) ])^ (l1[4]=l2[4])^...... ])^ (l1[k-2]=l2[k-2]) ])^ (l1[k-1]=l2[k-
1]) then{
        c=l1 join l2;
                if has_infrequent_subset(c,$L_{k-1}$) then
                delete c;
                else add c to $C_k$;
                }
        end for
end for
return $C_k$;
}


procedure has_infrequent_subset(c,$L_{k-1}$)
{
for each k-1 subset of s of c
        If S not belongs to $L_{k-1}$ then
        return TRUE;
return FALSE;
}
```

//Method 2//

```
For( i = 1; i < n ; i++ )
{
Read in partition $p_i$ ($p_i$ belongs to p);
$L^i$=gen_i_length_itemsets($p_i$);
f(x);//shifting partitions
f(y);//shifting  contents in pipeline stages to next stage
}
```

<u>Procedure for gen_i_length_itemsets(p_i)</u>

```
$L_1^p$={ large 1- itemsets along with their  tidlists }
For( k = 1 ; k ≤ m ; k++ )
{


For all itemsets $l_1$ € $L_{k-1}^p$ do begin


For all itemsets $l_2$ €  $L_{k-1}^p$ do begin
If ( $l_1[1]=l_2[1]$ □$l_1[2]=l_2[2]$ □……..$l_1[k-1]<l_2[k-1]$ ) then
{
c= $l_1[1].l_1[2] .l_1[3].l_1[4]$ ……..$l_1[k-1].l_2[k-1]$;
}
If c cannot be pruned then
c.tidlist=$l_1$.tidlist □$l_2$.tidlist
if( |c.tidlist|/|p|>=min. Sup.) then
$L_k^{p=}$ $L_k^p$ □ { c }
End
End
} return □$_k$ $L_k^p$
```

<u>For shifting partitions right (f(x))</u>

```
{int array[n]
for (i=n-1;i>=0;i--)
{
array [i]=array[i-1];
}}
```

<u>For shifting contents in pipeline stages to next stage (f(y))</u>

```
{int array[m]
for (i=m-1;i>=0;i--)
{
array [i]=array[i-1];
}}
```

X={pipeline,partition}

F(pipeline);(Now shifting algorithm for pipeline)

F (partition);(shifting of partition to pipeline)

5. DISCUSSION

The efficiency of our algorithm can be highly improved if it is performed on multiprocessor system and used divide & rule method. In the Apriori algorithm, the support plays a decisive role, but now the confidence will be in the first place to mine some association rules with very high degree of confidence. The entire algorithm is divided into three steps: generate optional term sets, filter optional large term sets, and generate association rules. In the three steps, the key is filtering candidate large term sets. After a first scan on the database, the initial optional term set is generated, and after the support is calculated ,the affair data term with low support is found, and then the confidence of it in the corresponding original database will be calculated through the method based on probability, if too high, the affair data term set which it belongs to will be filtered to reduce the number of records in next search; thus in the second scanning database, the scanning scope will be reduced and the searching time will be shorten, which can improve the algorithm efficiency by the same way, finally a large optional set will be generated, resulting association rules, which will be output.

6. CONSLUSION & FUTURE WORK

In this paper we described the data mining approach applied to the data cloud spread across the globe and connected to the web. We used the Sector/Sphere framework integrated to association rule based data mining. This enables the application of the association rule algorithms to the wide range of Cloud services available on the web. We have described a cloud-based infrastructure designed for data mining large distributed data sets over clusters connected with high performance wide area networks. Sector/Sphere is open source and available through Source Forge. The discovery of the association rule is a most successful and most vital duty in the data mining, is a very active research area in current data mining, its goal is to discover all frequent modes in the data set, and the current research work carrying on are mostly focused on the development of effective algorithm. On the basis of in-depth study of the existing data mining algorithms, in this paper a new data mining algorithm based on association rules is presented. The algorithm can avoid redundant rules as far as possible; the performance of the algorithm can be obviously improved when compared with the existing algorithms. We are implementing the algorithm and comparing our algorithm with other approaches.

7. REFERENCES

[1] Amazon. Amazon Simple Storage Service (Amazon S3). www.amazon.com/s3.

[2] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. “The Google File System”. In *SOSP*, 2003.

[3] Dhruba Borthaku. “The hadoop distributed file system: Architecture and design”. retrieved fromlucene.apache.org/hadoop, 2007.

[4] Jeffrey Dean and Sanjay Ghemawat. “MapReduce: Simplifieddata processing on large clusters”. In *OSDI’04: SixthSymposium on Operating System Design andImplementation*, 2004.

[5] Amazon Web Services LLC. “Amazon web services developer connection”. retrieved from developer.amazonwebservices.com on November 1, 2007.

[6] Hillol Kargupta. *Proceedings of Next Generation Data Mining2007*. Taylor and Francis, 2008.

[7] Ian Foster and Carl Kesselman. "*The Grid 2: Blueprint for a New Computing* infrastructure". Morgan Kaufmann, San Francisco, California, 2004.

[8] Yunhong Gu and Robert L. Grossman. "UDT: UDP-based data transfer for high-speed wide area networks". *Computer Networks*, 51(7):1777—1799, 2007.

[9] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H Balakrishnana. "Chord: A scalable peer to peer lookup service for internet applications". In *Proceedings of the ACM SIGCOMM '01*, pages 149–160, 2001.

[10] 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)

[11] Agrawal R. Imielinski T, Swami. "A Database mining: a performance perspective". IEEE Transactions on Knowledge and Data Engineering, Dec.1993,5(6): 914 - 925.

[12] Jim Gray and Alexander S. Szalay. "The world-wide telescope". *Science*, vol 293:2037–2040, 2001.

[13] Yunhong Gu and Robert L. Grossman. "UDT: UDP-based data transfer for high-speed wide area networks". *Computer Networks*, 51(7):1777—1799, 2007.

[14] 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.

[15] ZouLi & LiangXu, "Mining Association Rules in Distributed System", First International Workshop on Education Technology and Computer Science,. IEEE, 2009.

[16] A. Rosenthal et al. "Cloud computing: A new business paradigm for Biomedical information sharing "Journal of Biomedical Informatics, Elsevier ,43 (2010) 342–353 343.

[17] G. Stumme et al. "Web Semantics: Science, Services and Agents on the World Wide Web", Journal of WEB Semantics, Elsevier, 4 (2006) 124–143.

[18] John P. Hayes ,"Computer Architecture and Organizaton",3/e, McGraw-HILL INTERNATIONAL EDITIONS, Computer Science Series, pp- 275-292 (1998)