*IOSR Journal of Computer Engineering (IOSRJCE)*
*ISSN: 2278-0661, ISBN: 2278-8727Volume 8, Issue 4 (Jan. - Feb. 2013), PP 54-61*
*www.iosrjournals.org*

# High performance Cloud data mining algorithm and Data mining in Clouds

[1]Nandini Mishra, [2]Saurabh Sharma, [3]Ashish Pandey
*[1] Department of computer Science,Bhagwant University,India,*
*[2]Department of computer Science,Bhagwant University,India,*
*[3]Assistant Prof. Department of computer Science,Bhagwant University,India*

***Abstract:*** *We describe the design and implementation of a high performance cloud that we have used to archive, analyze and mine large distributed data sets. 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. High-performance can be reasonably intended as a intermediate step of high-performance data mining activities over large-scale amounts of data, while still keeping unaltered the primary and self-contained focus of achieving effectiveness and efficiency in these task themselves. 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. 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.*

***Keywords:*** *Sphere, Sector, Data mining, Cloud computing, Classification, Genetic algorithm, Cloud Model, High performance cloud.*

## I. 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 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. This paper describes a distributed high performance data mining system that we have developed 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 indexedles. Different segments of the 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 paral-lelism. Sector is designed to take advantage of wide area high performance networks when available. Sphere is designed to execute user dened functions in par- allel using a stream processing pattern for data managed by Sector. We mean by this that the same user de_ned function is applied to every data record in a data set man- aged by Sector. This is done to each segment of the data set independently (assuming that sucient processors are available), providing a natural parallelism. The design of Sector/Sphere results in data frequently being processed in place without moving it. To summarize, Sector manages data using distributed, indexedles; Sphere processes data with user dened functions that operate in a uniform manner on streams of data managed by Sector; Sector/Sphere scale to wide area high performance networks using specialized network protocols designed for this purpose. 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. 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:

- Providing a common syntax for machine understandable statements.
- Establishing common vocabularies.
- Agreeing on a logical language.
- 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. Cloud computing is one of the most explosively expanding technologies in the computing industry today. A Cloud computing implementation typically enables users to migrate their data and computation to a remote location with some varying impact on system performance. This provides a number of benefits which could not otherwise be achieved.
Such benefits include:

- *Scalability* - Clouds are designed to deliver as much computing power as any user needs. While in practice the underlying infrastructure is not infinite, the cloud resources are projected to ease the developer's dependence on any specific hardware.
- *Quality of Service (QoS)* - Unlike standard data centers and advanced computing resources, a well-designed Cloud can project a much higher QoS than traditionally possible. This is due to the lack of dependence on specific hardware, so any physical machine failures can be mitigated without the prerequisite user awareness.
- *Customization* - Within a Cloud, the user can utilize customized tools and services to meet their needs. This can be to utilize the latest library, toolkit, or to support legacy code within new infrastructure.
- *Cost Effectiveness* - Users finds only the hardware required for each project. This reduces the risk for institutions potentially want build a scalable system, thus providing greater flexibility, since the user is only paying for needed infrastructure while maintaining the option to increase services as needed in the future.
- *Simplified Access Interfaces* - Whether using a specific application, a set of tools or Web services, Clouds provide access to a potentially vast amount of computing resources in an easy and user-centric way.

## II. Background And Related Work

The problem we address lies at the intersection of data mining and high performance computing. Accordingly, we provide a survey of relevant work from both areas. Dataset sizes that exceed the memory capacity of a desktop computer pose a major challenge for data mining. This limitation can be mitigated through optimized algorithm design and the use of sampling or ensemble methods . With improvements in multi-processor machines, and more recently multicore technology, greater scalability can be achieved by effectively parallelizing algorithm implementations. But these approaches remain limited because (i) performance gains often cannot b realizedbeyond 8-16 cores due to communication overhead and (ii) dataset sizes are restricted to the total memory available in the system, generally on the order of a few Gigabytes.

***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.
- *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.
  - *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: i*mproves through the use of multiple redundant sites, which makes cloud computing suitable for business continuity and disaster recovery. Nonetheless, many major cloud computing.
  - *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.
- *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..

**2.1. Data Mining Algorithm**

Cloud computing, with its promise of virtually infinite computing and storage resources, is suitable to solve resource greedy computing problems. One problem of data mining in the cloud has been investigated from the data mining algorithm perspective. utilized the powerful and huge capacity of cloud computing into data mining and machine learning. In their experiments, three algorithms, i.e., global effect (GE), K-nearest neighbor (KNN) and restricted boltzmann machine (RBM) were performed in cloud computing platforms, which use the S3 and EC2 of Amazon Web Services. And they built two predictors based on KNN model and RBM model respectively with the order to testify their performance based on cloud computing platforms

**2.2. Architecture of Data Mining**

Cloud is an infrastructure that provides resources and services over the Internet. Generally speaking, a cloud computing platform consists of a storage cloud, a data cloud and a compute cloud, which are responsible for storage services, data management and computational tasks.

**2.3.Dm Algorithm Properties**

In this section we describe several key properties of efficient large-scale parallel DM systems addressed by the GraphLab abstraction and how other parallel frameworks fail to address these properties.

***2.3.1.Graph Structured Computation***: Many of the recent advances in DM have focused on modeling the *dependencies* between data. By modeling data dependencies, we are able to extract more signal from noisy data. For example, modeling the dependencies between similar shoppers allows us to make better product recommendations than treating shoppers in isolation.

.

***2.3.2.Asynchronous Iterative Computation*****:** Many important DM algorithms iteratively update a large set of parameters. Because of the underlying graph structure, parameter updates (on vertices or edges) depend (through the graph adjacency structure) on the values of other parameters Dynamic Computation: In many DM algorithms, iterative computation converges asymmetrically. For example, in parameter optimization, often a large number of parameters will quickly converge in a few iterations, while the remaining parameters will converge slowly over many iterations.

***2.3.3.Serializability****:* By ensuring that all parallel executions have an equivalent sequential execution, serializability eliminates many challenges associated with designing, implementing, and testing parallel DM algorithms. In addition, many algorithms converge faster if serializability is ensured, and some even require serializability for correctness.

## III. Methodologies And Model

**3.1 . Classification**

Classification is an important mission in data mining, and probably has become the most studied data mining task. In this The dataset is symbolized by an attribute set consists of a number of attributes, R=($a1$, $a2$,…, $a$N), where $ai$($i$=1,2,…,N) is an attribute. The attribute set can be divided into two parts: 1) predicting attributes, C=($c1$, $c2$,…, $c$m); 2) class attributes, D=($d1$, $d2$,…, $d$n).

A classification rule is demonstrated as:

If $(c1\in I1)\wedge(c2\in I2)\wedge\ldots\wedge(cm\in Im)$,
Then $(d1\in J1)\wedge(d2\in J2)\wedge\ldots\wedge(dn\in Jn)$
where I$i$ and J$j$ ($i$=1,…,m; $j$=1,…,n) are values of $ci$ and $dj$ respectively.

“ If” part contains a combination of conditions, and “Then” part contains the predicted class labels. The data record is horizontally divided into two parts – training set and test set – that are mutually exclusive and exhaustive. The data mining algorithm discovers classification rules based on the training set, and evaluates the predictive performance of these rules with the test set.

### 3.2. Genetic Algorithm

Genetic algorithm (GA) is a computational model inspired by evolution. This algorithm encodes a specific problem with several attributes on a simple chromosome-like data structure, and generates a population of chromosomes randomly as the initial colony. Then it applies operators - selection operator, crossover operator and mutation operator - to these structures to get an optimal solution evaluated by a fitness function. The workflow of GA is depicted as follow:

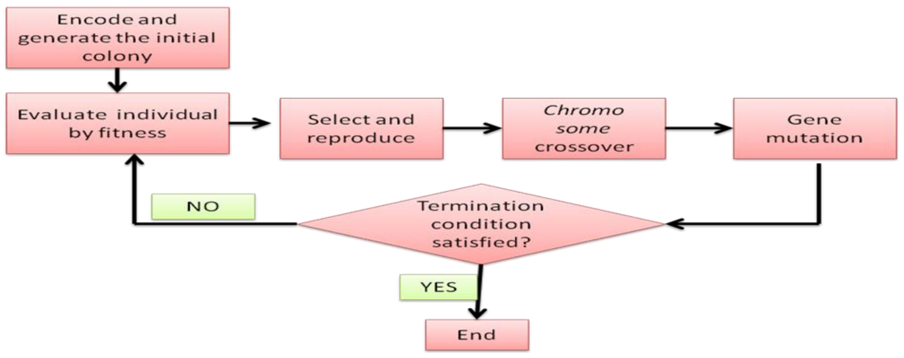
Encode and generate the initial colony
Evaluate individual by fitness
Select and reproduce
Chromo some crossover
Gene mutation
NO
Termination condition satisfied?
YES
End

**Fig.1 Genetic Algorithm**

### 3.3. Classification Rules Mining in Cloud Computing Environment

***3.3.1. Constraints analysis**:* Cloud computing is a network where distributed computer clusters constitute the resource pool of hardware. Tasks are divided into parallel segments, and assigned to available computing resources for processing. In that case, whether a computing task can be handled by cloud depends on decomposability and parallelism of the task. The requirements of processing computation tasks in cloud are defined as follow:

- The task is capable of being divided into mutually exclusive sub-tasks.
- Sub-tasks and data should be allocated to unoccupied processing nodes.
- Mechanism of synchronization and communication amongst processing nodes is indispensable.

***3.3.2. Model description:*** With storage and computing resources dispersing in the cloud environment, a Master is set as a central controller to search and distribute the resources. The responsibilities of Master are defined as: First, it divides the entitle task into several sub-tasks (here, a sub-task can be regarded as a data section), and assigns them to dispersed processing node; Afterwards, sub-tasks are executed on distributed nodes under the supervision of the Master; Eventually, the Master collects processing results on every node and synthesizes them to present a comprehensive set of classification rules as the mining result.
The pseudo code of programming the model is displayed as:

```
SegSet=Master.dataSeg(DateFile, N);
SegSet=(Seg(1), Seg(2),..., Seg(N));
For(i=1;i≤N;i++)
{PUData(i)=Master.copyData(Seg(i));
GAClassify(PUData(i));
//calculate classification rules in form of <key, value>
Input classy results into Buffer;}
For(j=1;j≤m;j++)
key(j).valuelist=Master.calculateValuelist(key(j));
//for each key in buffer, calculate a valueList consists of rules sharing this key.
For(j=1;j≤m;j++)
{CUData(j)=Master.distributeValuelist(ValueList(j));
GAClassify(CUData(j));
//calculate classification rules in form of <key, valueList>}
Master.getResults(CU)
```

### 3.4. *Algorithm Design*

The canonical GA is adapted to cloud computing environment for mining classification rules. However, it follows the basic procedure of canonical GA, including encoding, initialization, fitness assessment, selection, crossover and mutation.

***3.4.1 Encoding:*** The first step of GA is encoding that represents variables of individuals into bit strings. Pittsburgh and Michigan approaches can be applied to encode individuals. In the Pittsburgh approach each individual encodes a set of prediction rules, whereas in the Michigan approach each individual encodes a single prediction rule. When the task is classification, interaction among the rules is important.

***3.4.2 Initial population and fitness function**:* Pieces of data in the dataset are encoded into binary strings and denoted as the initial population. After creating the initial population, evaluate each string by a fitness function with the result of a fitness value.

***3.4.3 Selection operator:*** The "remainder stochastic sampling" which will more closely match the expected fitness values, is applied to conduct selection process.

***3.4.4 Crossover operator:*** After selection operation, crossover occurs. Crossover is recombination of paired strings in order to create new samples. According to the probability of *Pc*, a crossover point is randomly chosen. Then the fragments after the crossover point swap and produce the offsprings.

***3.4.5 Mutation operator**:* When crossover finished, it is time for mutation. Mutation is applied to each bit in the strings with a low probability denoted *Pm*. If a bit is selected to mutate, the bit value will change within its value range. After the process of selection, crossover and mutation, the next generation is formed.

## IV. Design

### 4.1. Design Of Sphere

***4.1.1 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. 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);
```

***4.1.2 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
.

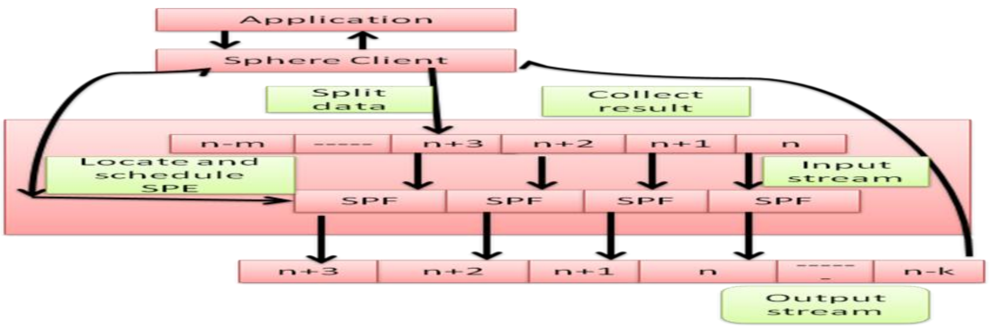

**Fig.2 The design of Sphere**

**4.2. Design Of Sector**

***4.2.1. 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.

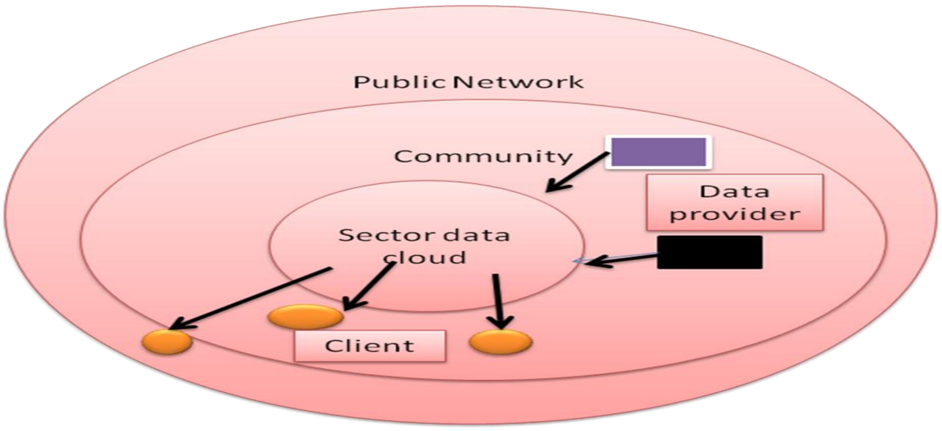

**Fig.3 The design of Sector.**

## V. Cloud Performance Evaluation

In this section we present a performance evaluation of cloud computing services for scientific computing One of the main concerns for enterprises that are considering cloud computing is performance. Achieving high-speed delivery of applications in the cloud is a multifaceted challenge that requires a holistic approach and an end-to-end view of the application request-response path .Performance issues include the geographical proximity of the application and data to the end user, network performance both within the cloud and in-and-out of the cloud and I/O access speed between the compute layer and the multiple tiers of data stores.

**Technology stack**Several cloud providers have focused their services on a particular software stack. This typically moves them from being Infrastructure as a Service (IaaS) providers to the realm ofPlatform as a Service (PaaS). As one would expect, the different stack-specific clouds align with the most popular software stacks out there.They can offer tremendous savings in terms of time and expense by shielding you from having to deal with lower level infrastructure setup and configuration. The flip side is that they often require developers to follow certain best practices in architecting and writing their apps, which creates a higher degree of vendor lock-in.

**5.1. Method** :We design a performance evaluation method that allows an assessment of clouds, and a comparison of clouds with other scientific computing infrastructures such as grids and PPIs. To this end, we divide the evaluation procedure into two parts, the first cloud-specific, the second infrastructure-agnostic.

***5.1.1.****Cloud-specific evaluation:* An attractive promise of clouds is that there are always unused resources, so that they can be obtained at any time without additional waiting time. However, the load of other large-scale systems varies over time due to submission patterns; we want to investigate if large clouds can indeed bypass this problem.

***5.1.2.****Infrastructure-agnostic evaluation* **:**There currently is no single accepted benchmark for scientific computing at large-scale . In particular, there is no such benchmark for the common scientific computing scenario in which an infrastructure is shared by several independent jobs, despite the large performance losses that such a scenario can incur . To address this issue, our method both uses traditional benchmarks comprising suites of job to be run in isolation and replays workload traces taken from real scientific computing environments.

**5.2 .Experimental Setup**

We now describe the experimental setup in which we use the performance evaluation method presented earlier.

***5.2.1*** *Performance Analysis Tool* :We have extended the GrenchMark large-scale distributed testing framework with new features which allow it to test cloud computing infrastructures. The framework was already able to generate and submit both real and synthetic workloads to grids, clusters, and other large-scale distributed environments. For this work we have added to GrenchMark the ability to measure cloud-specific metrics such as resource acquisition time and experiment cost.

***5.2.2. Cloud resource management:*** We have also added to the framework basic cloud resource management functionalities, since currently no resource management components and no middleware exist for accessing and managing cloud resources.

***5.2.3. Performance metrics*** **:**We use the performance metrics defined by the benchmarks used in this work. We also define and use the HPL efficiency for a real virtual cluster based on instance type T as the ratio between the HPL benchmark performance of the cluster and the performance of another real environment formed with only one instance, expressed as a percentage.

*5.2.4. Environment* :We perform all our measurements on the Amazon EC2 environment. However, this does not limit our results, as there are sufficient reports of performance values for all the Single-Job benchmarks .on the five EC2 instance types.

*5.2.5. Optimizations, tuning* :The benchmarks were compiled using command-line arguments. We did not use any additional architecture- or instance-dependent optimizations. For the HPL benchmark, the performance results depend on two main factors: the the Basic Linear Algebra Sub- program, and the problem size

**5.3. Experimental Results**

The experimental results of the Amazon EC2 performance evaluation are presented in the following.

**5.3.1 Resource Acquisition and Release**

We study three resource acquisition and release scenarios: for single instances over a short period, for multiple instances over a short period, and for single instances over a long period of time.

- Single instances: We first repeat 20 times for each of the five instance types a resource acquisition followed by a release as soon as the resource status becomes installed . the overheads associated with resource acquisition and release in EC2. The total resource acquisition time (Total ) is the sum of the Install and Boot times.
- Multiple instances: We investigate next the performance of requesting the acquisition of multiple resources at the same time; this corresponds to the real-life scenario where a user would create a homogeneous cluster from Amazon EC2 resources.
- Long-term investigation Last: we discuss the Install time measurements published online by the inde- pendent CloudStatus team . We have written web crawlers and parsing tools and taken samples every two minutes between Aug 2008 and Nov 2008 (two months).

***5.3.2. Performance of Single-Job-Single-Machine Workloads***

In this set of experiments we measure the raw performance of the CPU, I/O, and memory hierarchy using the Single-Instance benchmarks .

- Compute performance :We assess the computational performance of each instance type using the entire LMbench suite. The performance of int and int64 operations, and of the float and double float operations.
- I/O performance: We assess the I/O performance of each instance type with the Bonnie benchmarking suite, in two steps. The first step is to determine the smallest file size that invalidates the memory-based I/O cache, by running the Bonnie suite for thirteen file sizes in the range 1024 Kilo-binary byte (KiB) to 40 GiB the results of the rewrite with sequential output benchmark, which involves sequences of read-seek-write operations of data blocks that are dirtied before writing We analyze the I/O performance obtained for files sizes above 5GiB in the second step; summarizes the results. We find that the I/O performance indicated by Amazon EC2 corresponds to the achieved performance for random I/O operations .
- Memory hierarchy performance: We test the performance of the memory hierarchy using CacheBench on each instance type.
- Performance stability: The results obtained from single machine benchmarks are consistent for each instance type.
- Reliabilit:y We have encountered several system problems during the SJSI experiments.

***5.3.3 Performance of Single-Job-Multi-Machine Workloads:***

In this set of experiments we measure the performance delivered by homogeneous clusters formed with Amazon EC2 instances when running the Single-Job-Multi-Machine benchmarks..

- HPL performance: The performance achieved for the HPL benchmark on various virtual clusters based on the m1 HPCC performance To obtain the performance of virtual EC2 clusters we run the HPCC benchmarks on unit clusters comprising one.
- HPCC performance: To obtain the performance of virtual EC2 clusters we run the HPCC benchmarks on unit clusters comprising one instance, and on 16-core clusters comprising at least two instances.stance, and on 16-core clusters comprising at least two instances.
- Reliability: We have encountered several reliability problems during these experiments; the two most im-portant were related to HPL and are reproducible.

### *5.3.4 Performance of Multi-Job Workloads :*

We use a trace from a real system to assess the performance overheads due to executing complete workloads instead of single jobs in a virtual Amazon EC2 cluster. To this end, we replay on EC2 a part of the trace of our multi-cluster DAS3 grid system, which trace contains the jobs submitted to one of the DAS3 clusters. *Stable results and low overhead for the execution of concurrent jobs* For each instance type, we have found the individual makespan values obtained from repeated workload replays to be less than 1% different from the median workload makespan.

## VI. 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. The primary goal of this manuscript is to evaluate the viability of virtualization within HPC. After our analysis, the answer seems to be a resounding ”yes.” However, we also hope to select the best virtualization technology for such an HPC environment. In order to do this, we combine the feature comparison along with the performance results, and evaluate the potential impact within the FutureGrid testbed. We will extend this work with additional analysis of the other services offered by clouds, and in particular storage and network; how do they respond to to the combined stress of workloads with different characteristics and requirements that the diverse population of cloud users are supposed to incur in the future? We will also extend the performance evaluation with other real and synthetic applications, toward creating a performance database for the scientific community. we have first performed a comprehensive performance evaluation of a large computing cloud that is already in production. Then, we have compared the performance and cost of clouds with those of scientific computing alternatives such as grids and parallel production infrastructures. Our main finding is that the performance and the reliability of the tested cloud are low. Thus, the tested cloud is insufficient for scientific computing at large, though it still appeals to the scientists that need resources immediately and temporarily. Motivated by this finding, we have analyzed how to improve the current clouds for scientific computing, and identified two research directions which hold each good potential.

## 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] Y unhong 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] D. Chappell, “Introducing windows azure,” Microsoft, Inc, Tech. Rep., 2009.

[6] K. Keahey, R. Figueiredo, J. Fortes, T. Freeman, and M. Tsugawa, “Science clouds: Early experiences in cloud computing for scientific applications,” Cloud Computing and Applications, vol. 2008, 2008.

[7] I. Foster, T. Freeman, K. Keahy, D. Scheftner, B. Sotomayer, and X. Zhang, “Virtual clusters for grid communities,” Cluster Computing and the Grid, IEEE International Symposium on, vol. 0, pp. 513–520, 2006.

[8] R. Creasy, “The origin of the VM/370 time-sharing system,” IBM Journal of Research and Development, vol. 25, no. 5, pp. 483–490, 1981.

[9] Amazon elastic compute cloud,” [Online], http://aws.amazon.com/ec2/.

[10] K. Keahey, I. Foster, T. Freeman, and X. Zhang, “Virtual workspaces: achieving quality of service and quality of life in the Grid,” Scientific Programming, vol. 13, no. 4, pp. 265–275, 2005

[11] Bedra, A. 2010. “Getting started with Google App Engine and Clojure”, Internet Computing, 14, 85-88.

[12] Wang, L., Tao, J., Kunze, M., Castellanos, A. C., Kramer, D., and Karl, W. 2008. “Scientific cloud computing: early definition and experience”. In Proceeding of the 10th IEEE International Conference on High Performance Computing and Communications.

[13] Grossman, R. L., Gu, Y., Sabala, M., and Zhang, W. 2009. “Compute and storage clouds using wide area high performance network”, Future Generation Computer Systems, 25, 179-183.

[14] Freitas, A. A. 2002. “A survey of evolutionary algorithms for data mining and knowledge discovery”, Advances in Evolutionary Computation, 819-845.