Evaluating the performance and Scheduling the access requests in cloud storage by using CloudSim Toolkit

M.Haritha[1], N.J.Subashini[2]

1. *Research Scholar,* M. Tech cloud computing, Dept. Of Information Technology, SRM University, Chennai, India.

haritha.m513@gmail.com

2. *Assistant Professor,* Dept of Information Technology, SRM University, Chennai, India,
Subashini.n@ktr.srmuniv.ac.in

Abstract:

Cloud computing is a rising technology. In which virtual machine migration and dynamic resource allocation can place a main role to produce services to the user. The user will send their access requests to the constantly to the cloud when migration was working. So here we are going to schedule those access requests to ensure the performance of cloud storage system [7]. Our main goal is to implement the scheduling algorithm for migration of Hosts. So here we could establish the connection between system availability and resource availability to improve the performance of cloud storage. The scheduling algorithm can be implemented in CloudSim toolkit. It supports both system and behaviour modelling of cloud system components such as data centres (DC), Hosts, Virtual machines (VMs).

Key words: Cloud computing, CloudSim, Storage, Migration

1. Introduction:

Cloud computing is a general term used for storage and delivering the services over internet. The [4] cloud services are mainly categorised into three types of the services those are Infrastructure as a service (IaaS), Software as a service (SaaS), Platform as a Service (PaaS). The cloud can be deployed as public cloud, private cloud, hybrid cloud, community cloud. The service providers, who expert at provisioning, managing, and scaling services for multiple customers, are providing offerings based on IaaS in which the enterprise uses the pay-as-you-go compute infrastructure from the service provider. A cloud provided by a service provider is known as a public cloud.

The cloud environment will provide the different platform by creating the virtual machine that supports users in completing their jobs within a reasonable time and cost-effectively without sacrificing the QoS (Quality of service). The huge growth in virtualization and cloud computing technologies affects the increasing number of jobs that require the services of the VM. [18]Various types of scheduling algorithms that have been applied on load balancing and measured with different performance metrics to evaluate the performance. Most of the scheduling algorithms are developed to accomplish two aims. The first aim is to improve the QoS in executing the jobs and provide expected results on time. The second aim is to maintain efficiency for all jobs.

Cloud storage systems are one of the subsets of the cloud computing model, focused on the provisioning of on-demand storage services[7]. Such services can be provided by public cloud, private clouds. While these services provide benefits like availability, scalability, flexibility, there are many challenges that must be considered. Some of those challenges are related to the security of the cloud infrastructure because in cloud storage infrastructure place a crucial role. Now a day's every one stores there in cloud to access from anywhere.

The service models enterprises must consider their applications belong in the cloud and how to migrate them. [2]The migration will give the best possible results in the performance. It works as a backup for our system. If one host or VM is not responding at a time the migration will help means it will provide the available data. While performing the migration down time is not acceptable or it need to be minimized with both existing and migrated applications available always for users then the cloud capacity can be increased over time. [3]When the expert is performing the migration operation at a same time the user also will try to access the same application at a time.[8] In that time the user suppose to wait for more time to access their data or that server will send acknowledgment it will be shown like data UNAVALIABLE.

[9]If host migrations are successful means the previous host have to quite because of that problem only user can not access their data. So to avoid such a type of problems we need to schedule the Host and Virtual machine. To implement scheduling algorithm for host and VM is difficult to configure in real environment. In present system if one host fails it will create another host or it will start the same process from starting? So migration of large amount of device is taking more time. In existing system the host and VM selection is random. So, to enhance and to avoiding those problems we can implement our own algorithm in CloudSim toolkit.

[14]CloudSim toolkit will provide the generalised and extensible simulation framework that enables modelling, simulation and experimental of emerging cloud computing infrastructure (IaaS) and applications (SaaS) services. Detailed discussion about CloudSim will discuss in bellow. The main purpose of this paper is to provide an overview of scheduling system architecture for cloud storage environment and implementing migration policies in CloudSim.

In present working environment every user is trying to store their data in online so that they access that data from anywhere. Cloud storage becomes the most optimal selection for those huge files. There are some limitations with this system. Those are large number of VMs image size can't be loaded in the migration; [13] Utility loss is increasing as a single and global. To avoid these limitations we are providing proper scheduling system architecture and novel scheduling policies for storage so that the end user can access their data continuously.

2. Related work:

2.1. Cloud computing scheduling:

In earlier systems job scheduling mechanism based on queue system. It was not having any priority for requests. It just sends their requests into the queue and it executes in FIFO manner. So this process take more time to execute each task at the same time so, we need to schedule those tasks based on time, priority, size, etc.. The job scheduling is a method that assigns suitable jobs to the suitable task server. The scheduling process mainly considered two types. One is select the appropriate jobs and second is schedule tasks in this job. If the task allocation fails it may increase the efficiency decline, traffic on networks.

Proposed algorithm is near-optimal scheduling algorithm and [10]novel scheduling algorithm. The near optimal scheduling algorithm implements the polices that can explore the heterogeneous environment with multiple data centers and service providers. It considers the cost, energy, efficiency, Usage, workload. It depends upon the

IJSER

architecture. Novel algorithms will works for scheduling the utility loss for different categories of cloud computing environment. [11]The novel scheduling mechanism can describe the results in theoretically as well as practically.

2.2 Simulation:

[20]Simulation is a means of experimenting with a detailed model of a real system to determine how the system will respond to changes in its structure, environment, or underlying assumptions. Simulation is a powerful means by which process or existing processes may be designed, evaluated, and visualized without running the risks associated with conducting tests on a real system. Simulation plays powerful roles in the analysis of the current state as well as future state vision for the reengineered process. Simulation is used to understand and analyse the current state of a system as well as to vision the future state of reengineered system. Simulation provides a powerful means of generating suggestions for improving or innovating systems. By using simulation we can check supposition whether it is running true or false. By using simulation we can find bottlenecks and we can give different workload for testing the process.

2.3. CloudSim:

CloudSim is a toolkit (library) for simulation of cloud computing scenarios. CloudSim provides basic classes for describing data centers, virtual machines, applications, users, computational resources, and policies (e.g., scheduling and provisioning). [15]CloudSim provides infrastructure and software as a service. CloudSim is described by a case study involving dynamic provisioning of a software in the [6]intra networking cloud (federation of a cloud). Classes can be extended or replaced, new policies can be added and new scenarios for utilization can be coded. Think of it as the building blocks for your own simulated cloud environment.

It contain the abstract classes by extending those classes we can implement our scheduling algorithms and

Features:

- Support for modelling and simulation of large scale cloud computing infrastructure, including data centers on a single physical computing node.
- The availability of virtualization engine, which aids in creation and management of multiple, independent, and co-hosted virtualized services on a datacenter node.
- It is the enhanced version of GridSim toolkit[5].
- CloudSim toolkit supports Novel features[10].

2.4. Modeling the cloud:

Datacenter: Models the core infrastructure level services. It is contains a set of hosts that which is responsible for managing VMs during their life cycle.

Host: the component that represents a physical computing node in a cloud. It assigned a pre-configured processing capability, memory, storage and scheduling policy for allocating processing cores to virtual machines.

VM: Models Virtual machines. The host can simultaneously instantiate multiple VMs and allocate cores based on predefined processor sharing policies. Policies are like time shared policies and space shared policies.

VM Scheduler: Determines how the available CPU resources of virtual machine are divided among cloudlets. There are two types of policies

IJSER

are offered in Scheduler: those are space shared scheduler, Time shared scheduler.

Cloudlet: This is user side applications.[15] Models the cloud based applications services which are commonly deployed in the data centers. Every application has a pre assigned instruction length. Cloudlet is having attributes those are referred in mips(million instructions per second). Each Cloudlet can show their result in mips.

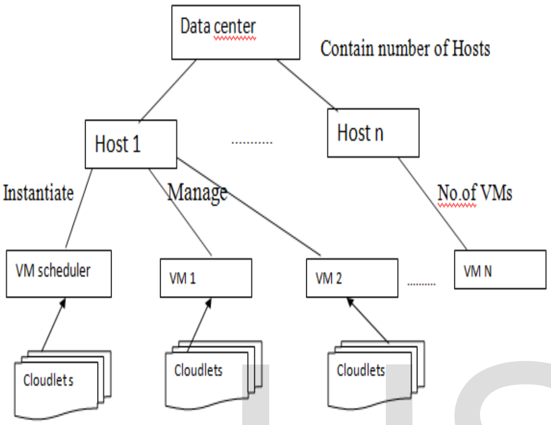

Fig 1: Cloud Infrastructure.

IJSER

2.5. How CloudSim works:

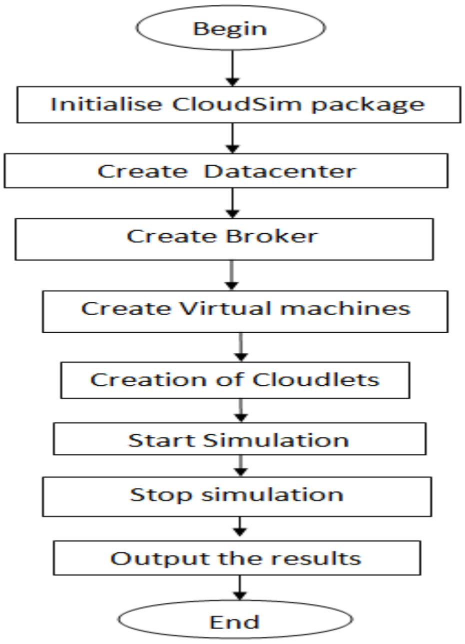

Fig 2: Basic working procedure of CloudSim.

The above diagram shows the basic working procedure of CloudSim. First step initialize the CloudSim package by using **init()** method that method will initialize the entities and variables of CloudSim. Second step is by using the base classes we can create Datacenter. Here we are creating datacenter virtually because CloudSim is a toolkit here we are analysing with our own configurations. After creating Datacenters it will construct Host with some attributes like id, Bandwidth, Storage, pelist (number of CPU), type of scheduler. Then crate broker. [17]Broker is actually an interface between a user and provider. Then VM and then cloudlet which are submitted by the user. The start the simulation by calling the function start Simulation after start the simulation it will start working and loading after completing all the jobs we have to stop the simulation step by step so by using stop Simulation we are going to stop. It shows the output in command prompt with each and every process.

3. Implementation:

3.1. Cloud environment Setup:

CloudSim is a toolkit for simulation of Cloud computing scenarios. The result of CloudSim significantly improves the application QoS requirements under fluctuate the resource and service demand patterns. It provides basic classes for describing data centers, virtual machines, applications, users, computational resources, and policies for management of diverse parts of the system (e.g., scheduling and provisioning). These components can be put together for users to evaluate new strategies in utilization of Clouds. It can also be used to evaluate efficiency of strategies from different perspectives, from cost/profit to speed up of application execution time. It also supports evaluation of Green IT policies.

3.2. Load dataset:

Data sets can be used as a unit to access the collection of shared data. In CloudSim there is no physical file system. Here we are using only the specific format of single large files. Those files can refer as object that can display as data sets. In that each file contains attributes like as name, size of a file, file owner, file created date, time, file updated time. In this module here we are using real time streaming data. That data can be chosen by user. That data can be written in size. By using space shared scheduling mechanism we are loading data. The loaded can be stored in queue. If process stopped because any uninterrupted service that will load from the previous end point. So that time will reduce to load data.

3.3. Allocation Jobs:

In this work here first we are analysing the each and every task which is known as cloudlet. The job is analyzed by using the time and last arrival of new data. So each job is allocation will depends on time shared or space shared policies. So finally the allocation of job depends on the deadline of the task.

3.4. Scheduling tasks:

[18]The scheduler filters out all the nodes that do not have enough resources available to host the new VM. Then, it iterates over all the remaining nodes, computing the content similarity between the new data node and all the data nodes running on the host nodes. It selects the node hosting the VM with the highest content similarity. Compared to the dedicated nodes scheduler, this approach is more computationally intensive, because many more nodes are evaluated to find the highest content similarities. The algorithm can be adjusted to limit the number of nodes it inspects, so that it does not need to inspect all the nodes in the system.

Virtual machines are migrating for maximizing the average resource utilization in cloud computing. So the migration of resource set is likely to occur in scheduling. Virtual machine which forms one VDN (virtual data node) may be migrated to another resource set before scheduler responses to end-users. Suppose that scheduler gives no response to end-users when virtual machine migrates. Therefore, scheduler must find, or make in some situation, the other new VDN transparently for that request. Moreover, scheduler must ensure that the performance of the whole request queue is affected as little as possible.

4. Overview of scheduling system:

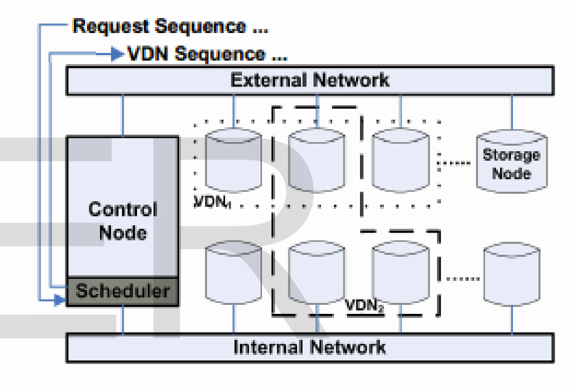

Fig 3: Overview of scheduler in cloud Storage.

The above diagram can show the scheduler for cloud storage. [19]Here we are using the distributed data storage node. The data can be transferred from VDN. The all VDN are virtualized by the resource nodes. The scheduler can return the sequence of data node in storage node. When the VM are migrating for maximizing the average resource utilization in cloud computing. The virtual machine forms the VDN that may be migrated to another resource before responding by the scheduler. The VDN transparently send the request to the other VDN for that request. Moreover scheduler can ensure the performance of the all requests in queue is affected as little as possible.

IJSER

4.1. Scheduling algorithm:

We suppose that there are n requests in the queue. All the data blocks that satisfied these requests are located in m VDNs. All these available VDNs are found via associative broadcast in Given that the response for each request contains k(i) data blocks. And each VDN has a fixed energy consumption and cost according to its storage and computing capacity. The problem is defined formally as below.

We are given a n x m matrix a. In this matrix, row is for request, and column is for VDN. a_{ij} represents the amount of data blocks on VDN_j which are satisfied request R_i. So,

$$\sum aij€aij \ a_{ij} = k(i)$$

According to matrix a, we define the other 3 matrix: waiting time T, cost C and energy consumption E.

In T, tij represents the amount of time when aij data blocks have been prepared for request Ri from VDNj . Since VDNs work in parallel, so the waiting time of request Ri is

$$\max \{tij|tij€ti\}$$

In C, C_{ij} represents the amount of cost that VDNj spends on to process aij data blocks for request Ri. The total cost for request R_i is:

$$\sum^{m}{}_{j\text{-}1} \ C_{ij}$$

In E, e_{ij} represents the amount of energy consumption that VDNj consumes to process aij data blocks for request Ri. The total energy consumption for request Ri is:

$$\sum^{m}{}_{j\text{-}1} \ e_{ij}$$

We have to balance the metrics of waiting time, cost and energy consumption for each Ri. We define $\dot{\omega}t$, $\dot{\omega}c$, $\dot{\omega}e$ to represent the weight of waiting time, cost and energy consumption. And

$$\dot{\omega}_{t+} \ \dot{\omega}_{c+} \ \dot{\omega}_{e=1}$$

The total overhead for responding Ri is

$$c_i = \dot{\omega}_t \ x \ \max_{tij€ti}\{t_{ij}\} + \dot{\omega}_c X \sum^{m}{}_{j=1} C_{ij} + \dot{\omega}_e \ x \sum^{m}{}_{j=1} e_{ij}$$

And the utility for request Ri is denoted as following:

$$u_i(a_i) = 1/C_i$$

For a single request Ri, the objective of scheduling is to maximize utility. The utility for the whole request queue is denoted as following:

$$u(R)=\sum u_i(a_i)$$

The above expression is the global utility function for a request queue. The goal of this scheduler is to maximize this global utility.

4.1.1. Minimization of Migrations VMs (MM)

```
Input: hostList Output: migrationList
foreach h in hostList do
vmList←h.getVmList()
vmList.sortDecreasingUtilization()
hUtil←h.getUtil()
bestFitUtil←MAX
while hUtil > THRESH_UP do
foreach vm in vmList do
if vm.getUtil() > hUtil − THRESH_UP then
t←vm.getUtil() − hUtil + THRESH_UP
if t < bestFitUtil then
bestFitUtil←t
bestFitVm←vm
else
if bestFitUtil = MAX then
bestFitVm←vm
break
hUtil←hUtil − bestFitVm.getUtil()
migrationList.add(bestFitVm)
vmList.remove(bestFitVm)
if hUtil < THRESH_LOW then
migrationList.add(h.getVmList())
vmList.remove(h.getVmList())
return migrationList
```

5. Conclusion and Future work:

In this paper we gave architecture for scheduling the requests in cloud storage. The scheduling algorithm that can given some better performance compare to the present working

IJSER

environment. The CloudSim will provide an output in command prompt. It will show the execution time and performance analysis results. By using some open issues of migration and storage we proposed a novel scheduling algorithm. Here we defined algorithm that depends on mainly two open issues those are [13]Utility loss for single request and global utility loss. The more detailed discussions and experimental results we are providing this algorithm. In future we are going to implement this CloudSim all scheduling algorithms and policies in CloudAnalyst tool kit and providing dynamic creation of virtual machines and host.

References:

1. **http://www.sciencedirect.com/science/article/pii/S2212671612001515#**
2. **http://callofduty.wikia.com/wiki/Host_Migration**
3. **http://community.callofduty.com/thread/200709939**
4. **http://south.cattelecom.com/rtso/Technologies/CloudComputing/0071626948_chap01.pdf**
5. **http://www.cloudbus.org/merit_flyers/GridSim-1page.pdf**
6. **http://www.cloudbus.org/papers/InterCloud-Brokering-Taxonomy.pdf**
7. **http://computer.howstuffworks.com/cloud-computing/cloud-storage3.htm**
8. **http://thecustomizewindows.com/2011/08/cloud-storage-unsolved-problems/**
9. **http://www.racemi.com/introvideo/**
10. **http://arxiv.org/ftp/arxiv/papers/0903/0903.2525.pdf**
11. **http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=6024605&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D6024605**
12. **http://searchdisasterrecovery.techtarget.com/High-availability-guidelines-and-VMware-HA-best-practices**
13. **http://mesl.ucsd.edu/site/pubs/Adnan_DATE_2013.pdf**
14. **https://code.google.com/p/cloudsim/**
15. **http://www.cse.sc.edu/~huhns/csce526/CloudSim.pdf**
16. **http://cloudbus.org/cloudsim/**
17. **http://www.cloudbus.org/papers/InterCloud-Brokering-Taxonomy.pdf**
18. **http://www.techrepublic.com/resource-library/whitepapers/advantages-challenges-and-optimizations-of-virtual-machine-scheduling-in-cloud-computing-environments/**
19. **http://www.djks123.cn/ieee_cd/data/papers/12014.pdf**
20. **https://engineering.purdue.edu/~engelb/abe565/week13.htm**

IJSER