Analysis and Improvement of Load Balancing in Cloud Computing

Ojasvee Kaneria
M.Tech Scholar
University College of Engineering,RTU,
Kota, Raj. India
Ojasveekaneria1@gmail.com

R K Banyal
Associate Professor
University College of Engineering,RTU,
Kota, Raj. India
Rkbayal@gmail.com

ABSTRACT
Cloud Computing is a form of distributed computing that makes it more proficient and accessible and useful. Implementation of cloud uses various phases, from inception to implementation, from usage to maintenance. The effective use of cloud depends on various aspects such as security, speed, privacy etc. Our main focus here is to make effective usage of cloud resources and to increase the access speed by modifying the basic load balancing algorithms. The allocation of virtual machines to the user bases is also improved and verification is done using CloudSim toolkit provided by CloudBus

Keywords
Cloud computing, load balancing, Task Scheduling, Round Robin, Throttled, Equal Load Sharing, CloudSim

1. INTRODUCTION
Cloud computing has made a powerful footstep in the IT Industry. Any business organization can access their data and resources from anywhere and at any time. This technology permits a way to cheaper computing. Cloud model shares the application resources area units between personal and public cloud. Various tools are developed tools to share workloads on cloud to manage peak conditions[1]. but sometimes they would like all environments to be supported same platform.
The current effective execution of cloud requires string hold on programming.
Programming refers for the connected set with policies to be able to manage the order of operations. From the inception to implementation, from usage to security, from speed to access various algorithms play there roles. The requests from the user bases forwarded to appropriate data centers and further allotment of virtual machines to these requests requires uses of various refined algorithms. Programming manages intelligence involving processor memory and smart programming policy provides most utilization of resource.

Cloud computing provides various mechanisms to target run time demand of computing resources, like storage, accessibility, package etc.

The on the market Cloud platforms distinguish among the service kind, the cost, the quality of Service (QoS) what is more as performance.

Development of economical service provisioning policies is that the foremost issues in Cloud analysis. The Cloud computing preponderantly offers three kinds of services Infrastructure as a Service (IaaS), Platform as a Service(PaaS), and package as a Service (SaaS) [2]. The various parameters that can be considered in cloud analysis are:[3]

- **Central Processing**
  The overall proportion of the CPU time was utilized or not.
- **Throughput**: Total no. of tasks dead (or requests served) per unit time.
- **Response Time**: The time spent within the waiting queue until it gets the primary time to use the central processing unit.
- **Waiting Time**: the overall time spent by the request waiting within the prepared queue once the primary response from central processing unit arrives.
- **Turnaround Time**: the overall time taken for latency, waiting time and repair time.
- **Resource Cost**: The overall price of the resources non-inheritable or used for the coupling of requests by varied

This paper aims to analyze various load balancing algorithms and propose a improved algorithm that aims maximum utilization and distribution of resources viz data centers and processing elements and hence reduces the request processing time.
The next section of the paper describes existing load balancing algorithms and shows the implementation of the result using CloudSim and Cloud analyst.
The next section describes the tools used followed by the proposed algorithm which is an improved version of existing algorithms for virtual machine provisioning and load balancing such as Round Robin and Throttled Load balancing. The following section is the results and experiment section followed by future scope and conclusion.

978-1-5090-5515-9/16/$31.00 ©2016 IEEE

Introduction to CloudSim [17]

Cloud computing is a recent advancement wherein IT infrastructure and applications are provided as 'services' to end-users under a usage-based payment model. It can leverage virtualized services even on the fly based on requirements (workload patterns and QoS) varying with time. The application services hosted under Cloud computing model have complex provisioning, composition, configuration, and deployment requirements. Evaluating the performance of Cloud provisioning policies, application workload models, and resources performance models in a repeatable manner under varying system and user configurations and requirements is difficult to achieve. To overcome this challenge, we propose CloudSim: an extensible simulation toolkit that enables modeling and simulation of Cloud computing systems and application provisioning environments. The CloudSim toolkit supports both system and behavior modeling of Cloud system components such as data centers, virtual machines (VMs) and resource provisioning policies. It implements generic application provisioning techniques that can be extended with ease and limited effort. Currently, it supports modeling and simulation of Cloud computing environments consisting of both single and inter-networked clouds (federation of clouds). Moreover, it exposes custom interfaces for implementing policies and provisioning techniques for allocation of VMs under inter-networked Cloud computing scenarios. Several researchers from organizations, such as HP Labs in U.S.A., are using CloudSim in their investigation on Cloud resource provisioning and energy-efficient management of data center resources.

The usefulness of CloudSim is demonstrated by a case study involving dynamic provisioning of application services in the hybrid federated clouds environment. The result of this case study proves that the federated Cloud computing model significantly improves the application QoS requirements under fluctuating resource and service demand patterns.

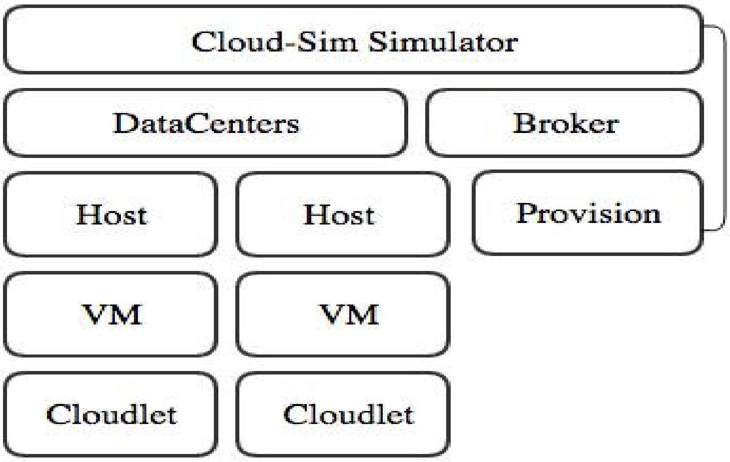

Figure1. Block diagram of Components of Cloud System[17]

CloudSim involves:

1. Datacenter: Set of rules called datacenters. This virtual model (VMS) (for example, could also be accountable in addressing VM provisioning.
2. Datacenter Broker: This class models the core infrastructure-level services (hardware) that are offered by Cloud providers (Amazon, Azure, App Engine). It encapsulates a set of compute hosts that can either be homogeneous or heterogeneous with respect to their hardware configurations (memory, cores, capacity, and storage). Furthermore, every Datacenter component instantiates a generalized application provisioning component that implements a set of policies for allocating bandwidth,memory, and storage devices to hosts and VMs.
3. Host: a decent host is connected to the information center. Ideas will host virtual machines.
4. VM: This class models a VM, which is managed and hosted by a Cloud host component. Every VM component has access to a component that stores the following characteristics related to a VM: accessible memory, processor, storage size, and the VM's internal provisioning policy that is extended from an abstract component called the C*loudletScheduler*
5. Cloudlet: This class models the Cloud-based application services. It orchestrates the complexity of an application in terms of its computational requirements. Every application service has a pre-assigned instruction length and data transfer (both pre and post fetches) overhead that it needs to undertake during its life cycle. This class can also be extended to support modeling of other performance and composition metrics for applications such as transactions in database-oriented applications.

Modeling the cloud[17]

The infrastructure-level services (IaaS) related to the clouds are simulated by extending the data center entity of CloudSim. The data center entity manages a number of hosts. The hosts are assigned to one or more VMs based on a VM allocation policy that should be defined by the Cloud service provider. Here, the VM policy stands for the operations control policies related to VM life cycle such as: provisioning of a host to a VM, VM creation, VM destruction, and VM migration. Similarly, one or more application services can be provisioned within a single VM instance, referred to as application provisioning in the context of Cloud computing.

Host is a CloudSim component that represents a physical computing server in a Cloud: it is assigned a pre-configured processing capability (expressed in millions of instructions per second—MIPS), memory, storage, and a provisioning policy for allocating processing cores to VMs. The Host component implements interfaces that support modeling and simulation of both single-core and multi-core nodes. VM allocation (provisioning) [7] is the process of creating VM instances on hosts that match the critical characteristics (storage, memory), configurations (software environment), and requirements (availability zone) of the SaaS provider. CloudSim supports the development of custom application service models that can be deployed within a VM instance and its users are required to extend the core Cloudlet object for implementing their application services. Furthermore, CloudSim does not enforce any limitation on the service models or provisioning techniques that developers want to implement and perform tests with. Once an application service is defined and modeled, it is assigned to one or more pre-instantiated VMs through a service-specific allocation policy.
Allocation of application-specific VMs to hosts in a Cloud-based data center is the responsibility of a VM Allocation controller component (called VM Allocation Policy). This component exposes a number of custom methods for researchers and developers who aid in the implementation of new policies based on optimization goals (user centric, system centric, or both). By default, VM Allocation Policy implements a straightforward policy that allocates VMs to the Host on a First-Come-First-Serve (FCFS) basis. Hardware requirements, such as the number of processing cores, memory, and storage, form the basis for such provisioning. Other policies, including the ones likely to be expressed by Cloud providers, can also be easily simulated and modeled in CloudSim. However, policies used by public Cloud providers (Amazon EC2, Microsoft Azure) are not publicly available, and thus a pre-implemented version of these algorithms is not provided with CloudSim.
For each Host component, the allocation of processing cores to VMs is done based on a host allocation policy. This policy takes into account several hardware characteristics, such as number of CPU cores, CPU share, and amount of memory (physical and secondary), that are allocated to a given VM instance. Hence, CloudSim supports simulation scenarios that assign specific CPU cores to specific VMs (a space-shared policy), dynamically distribute the capacity of a core among VMs (time-shared policy), or assign cores to VMs on demand.
Each host component also instantiates a VM scheduler component, which can either implement the space-shared or the time-shared policy for allocating cores to VMs. Cloud system/application developers and researchers can further extend the VM scheduler component for experimenting with custom allocation policies.

3. LOAD BALANCING ALGORITHMS

CloudSim implements following load balancing algorithms by default[4,5,6,7,8].

1. Round Robin
2. Throttled

Round Robin It is one of the simplest scheduling technique that utilize the principle of time slices. Here the time is divided into multiple slices and each node is given a particular time slice or time interval i.e. it utilizes the principle of time scheduling. Each node is given a quantum and its operation. The resources of the service provider are provided to the requesting client on the basis of time slice.

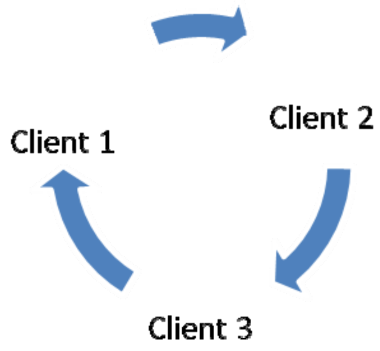

Figure2. Block diagram of RR Algorithm

Throttled : This algorithm ensure that pre-defined number of cloudlets are allocated to a single VM at any given time. If there are more request groups are present than the number of available VM's at data centre allocate incoming request in queue basis until the next VM becomes available.

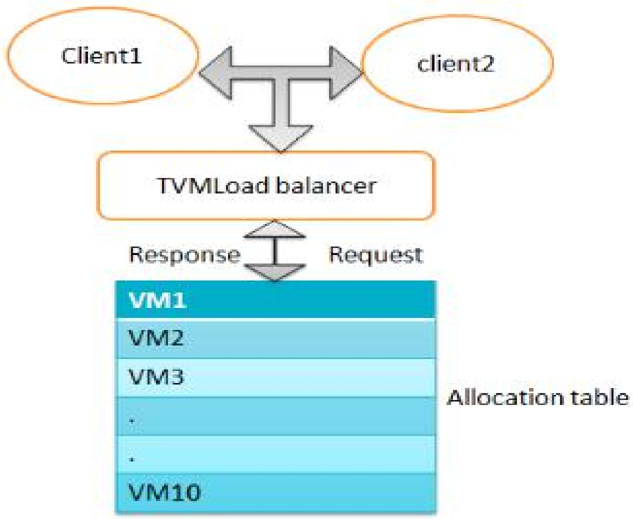

Figure2. Block diagram of Throttled Algorithm

EXISTING SYSTEM

Existing load balancing algorithms focus on allocation of data centers Few algorithms check on availability of free hosts if no free hosts are available it passes on to the next data center. Considering that the work arrival pattern is not predictable a dynamic theme may be used here for its freedom.[9,10,13]

6. PROPOSED ALGORITHM

The algorithm checks if there are any free hosts in the data center and if there are any then it assigns a process to that host and decreases the number of free processors with that host. If there are no free hosts, but have some free processors then it checks the host with the maximum number of processors and assigns the new incoming process to that host.

We define following procedure for allocation of host for virtual machine.

Proposed System performs the following steps in proposed VM provisioner:

```
      Procedure MyProvisioner
      {
      .
      .
  List<Integer> freePesTmp = new ArrayList<Integer>();
      for (Integer freePes : freePEs.clone())
             {
              freePesTmp.add(freePes);
             }
  If(VmTable contains VmId and UserId() {
                 Repeat
             moreFree = Integer.MIN_VALUE;
         for (i < freePesTmp.size())
         {
if((freePesTmp.get(i)>moreFree)&& (tempstatus[i]!=1))
      {
        moreFree = freePesTmp.get(i);
         idx = i;
            }
         tempstatus[idx]=1;
        for(j< freePesTmp.size()
              {
              if(tempstatus[j]==1)
                 flag=1;
               else
                 flag=0;
               }
               if(flag==1){
              moreFree = Integer.MIN_VALUE;
        for (k < freePesTmp.size())
       {
      if (freePesTmp.get(k) > moreFree) {
        moreFree = freePesTmp.get(
      idx = k;
      Host host = (Host)resources.get(idx);
      result = host.vmCreate(vm);
               if (result) {
      vmTable.put(vm.getVmId()+"-
      "+vm.getUserId(),host);
      usedPEs.put(vm.getVmId()+"-
      "+vm.getUserId(),requiredPes);
      freePEs[idx]-=requiredPes;
      result=true;
                            }
                      else {
      freePesTmp.set(idx, Integer.MIN_VALUE);
               tempstatus[idx]=0;
                            }
               tries++;
          }while(!result && tries<freePEs.length);
                 }
                 return result;
              Procedure Ends
```

7. EXPERIEMENT & RESULTS

Proposed system implement in java language with cloud sim simulator. Created cloudlets, VMs, Datacenters and Host with specific configuration. After create cloudlets request for VM on Datacenter. We also design resources allocation policy for CloudSim simulator. Resources Allocation policy describe how resources allocated to cloudlets. Configuration parameters show in below table.

Results of the Simulation Completed at: 18/10/2016 09:25:38

Overall Response Time Summary

	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	159.01	37.61	364.64
Data Center processing time:	0.38	0.02	0.89

Response Time by Region

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	50.19	37.61	60.11
UB2	49.99	38.64	59.89
UB3	49.98	38.88	61.06
UB4	300.72	234.14	364.64
UB5	300.34	225.14	355.64
UB6	199.97	155.11	242.11

Fig:4 Results of Proposed Algorithm

Results of the Simulation Completed at: 18/10/2016 09:23:32

Overall Response Time Summary

	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	159.02	37.61	364.64
Data Center processing time:	0.38	0.02	0.89

Response Time by Region

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	50.21	37.61	60.11
UB2	50.00	38.64	59.89
UB3	49.94	38.88	61.06
UB4	300.75	234.14	364.64
UB5	300.34	225.14	357.14
UB6	200.01	155.11	242.11

Fig: 5 Results of Existing RR Algorithm

8. CONCLUSION

This paper proposes an efficient algorithm for distribution of task on the respective data centers. It checks for the availability of hosts on the data centers as well as the processing elements and allocates the task accordingly. Till date not much work has been done identifying the availability of the PE's, our aim is to identify the available data centers and recursively search for the availability of processing elements. The experiments performed prove that implementation of the algorithm in VM provisioner effects the results in a positive way.

9. REFERENCES

[1] Ritu Kapur "A Cost Effective approach for Resource Scheduling in Cloud Computing" IEEE International Conference on Computer, Communication and Control (IC4-2015).

[2] Aditya Makwe , Priyesh Kanungo"Scheduling in Cloud Computing Environment Using Analytic Hierarchy Process Model" IEEE International Conference on Computer, Communication and Control (IC4-2015).

[3] Bhatiya Wickremansinghe1, Rodrigo N. Calheiros2and Dr. Rajkumar Buyya3, "CloudAnalyst: A CloudSim-based Visul Modeller for Analysing Cloud Computing Environments and Applications", IEEE Computer Society, 2010, pp. 446-452

[4] Kunal Mahurkar1, Shraddha Katore2 and Suraj Bhaisade3, Pratikawale4, "Reducing Cost of Provisioning in Cloud Computing", International Journal of Advance in Computer Science and Cloud Computing, Volume- 1, Issue- 2, nov.- 2013, pg. 6- 8.

[5] Pooja Samal1 and Pranati Mishra2, "Analysis of Variants in Round Robin Algorithms for Load Balancing in Cloud Computing", (IJCSIT) International Journals of Computer Science and Information Technologies, Volume 4 (3), 2013, pg. no. 416- 419.

[6] Jaspreet Kaur, "Comparison of load balancing algorithm in a Cloud", International Journal of Engineering Research and Applications (IJERA), vol. 2, Issue 3, May- June 2012, pp. 1169- 1173.

[7] G. Khanna, K. Beaty, G. Kar, and A. Kochut, "Application Performance Management in Virtualized Server Environm," in Network Operationsand Management Symposium, (2006). NOMS (2006). 10th IEEE/IFIP, 2006, pp 373–381.

[8] B. Santosh Kumar1 and Dr. Latha Parthiban2, "An Implementation of Load Balancing Policy for Virtual Machines Associated with a Data Centre", International Journal of Computer Science & Engineering Technology (IJCSET), volume 5 no. 03, March 2014, pp. 253- 261.

[9] Sonika Matele1, Dr, K James2 and Navneet Singh3, "A Study of Load Balancing Issue Among Multifarious Issues of Cloud Computing Environment", International Journals of Emerging Technolog Computational and Applied Science (IJETCAS), volume 13- 142, 2013, pg. 236- 241.

[10] Kunal Mahurkar1, Shraddha Katore2 and Suraj Bhaisade3, Pratikawale4, "Reducing Cost of Provisioning in Cloud Computing", International Journal of Advance in Computer Science and Cloud Computing, Volume- 1, Issue- 2, nov.- 2013, pg. 6- 8.

[11] Dr Hemant S. Mahalle1, Prof Parag R. Kaver2 and Dr. Vinay Chavan3, "Load Balancing on Cloud Data Centres", Internatinal Journal of Advanced Reserch in Computer Science and Software Engineering, volume 3, issue 1, January 2013, pp. 1- 4.

[12] Dr. Rajkumar Buyya, Rajiv Ranjan and Rodrigo N. Calheiros, "Modeling and Simulation of Scalable Cloud Computing Environments and the CloudSim Toolkit: Challenges and Opportunities", published online 24 august in Wiley Online Library Volume 41, Issue 1 January 2011 Pages 23–50.

[13] Deepali Tripathi; N. K. Joshi "Model for heterogeneous data integration on cloud" 2016 3rd IEEE International Conference on Computing for Sustainable Global Development (INDIACom) 2016 Pages 1307-09.

[14] Dr. Rajkumar Buyya, "CloudSim: a toolkit for modelling and simulation of cloud computing environments and evaluation of resource provisioning algorithm", published online 24 august in Wiley Online Library 2010, pp. 23- 50