ISSN:2320-0790

An International Journal of Advanced Computer Technology

A Cost Effective Approach for Provisioning In Cloud

Manisha Ghorpade[1], Mangesh Wanjari[2]

[1]CSE Department, RCOEM, Nagpur (MS), India

[2]CSE Department, RCOEM, Nagpur (MS), India

Abstract: Cloud computing allows business customers to scale up and down their resource usage based on their requirements. Many of the hyped gains in the cloud model are derived from resource multiplexing through virtualization. A load balancer which calculates the load on individual virtual server can be used to divert the web session load from server to server to enhance the response time to individual clients. By shifting multiple servers we can save the energy and also give advantage of dynamic resource allocation to the cloud users. Hence, we are proposing the system that uses collection of Virtual Machines (VMs) running on unused computers at the edge and uses virtualization technology to allocate resources dynamically based on application demands. We are trying to implement a set of heuristics that prevent overload in the system. The concept of parallel computing is used in which master machine or server allocates task to all connected (available) client machines and acknowledgement given back to the master after completion of the task. Master also keeps track of load balancing and failure management.

Keywords: Resource monitoring, Connection establishing, Load balancing, parallel computing.

I. INTRODUCTION

The state-of-art technology focuses on data processing to deal with massive amount of data. Cloud computing is a popular Buzzword today. It is a technology, which enables one to accomplish the aforementioned objective, leading towards improved business performance. It comprises of users requesting for the services of diverse applications from various distributed virtual servers. The cloud should provide resources on demand to its clients with high availability, scalability and with reduced cost. Load balancing is one of the essential factors to enhance the working performance of the cloud service provider. Since, cloud has inherited characteristic of distributed computing and virtualization there is a possibility of occurrence of deadlock. Hence, in this paper, a load balancing algorithm has been proposed to avoid deadlocks among the Virtual Machines (VMs) while processing the requests received from the users by VM migration. Further, this paper also provides the anticipated results with the implementation of the proposed algorithm. The deadlock avoidance enhances the number of jobs to be serviced by cloud service provider and thereby improving working.

The success of IT organizations lies in acquiring the resources on demand. Cloud computing is a promising technology to provide on demand services according to the clients requirements within a stipulated time. Further, the cloud computing environment provides the users for accessing the shared pool of distributed resources. Cloud is a pay- go model where the consumers pay for the resources utilized instantly, which necessitates having highly available resources to service the requests on demand. Hence, the management of resources becomes a complex job from the business perspective of the cloud service provider. Further, there can be a scenario where in the cloud service provider's datacenter will be hosting less number of Virtual Machines (VMs) compared to the number of jobs arrived for availing the service. In such a situation, similar types of jobs will be competing to acquire the same VM at the same time leading to a deadlock. Further, the deadlock problem leads to the degradation of working performance as well as the business performance of the cloud service provider. Henceforth, an efficient load balancing technique is required to distribute the load to avoid deadlocks.

To achieve the above mentioned objective a load balancing algorithm that supports migration has been proposed in this paper. In the cloud computing environment the load refers to the number of requests that has to be serviced by VMs that are available in cloud. The proposed algorithm avoids the deadlock by providing the resources on demand resulting in increased number of job executions. Henceforth, the business performance of the cloud service provider is improved by reducing the rejection in the number of jobs submitted. In order to implement the proposed technique hop time and wait time may be considered. Hop time is the duration involved in migration of the job from the overloaded VM to the underutilized VM for providing the service. Wait time is the time after which the VMs become available to service the request.

The main objective of the system is to shift multiple servers on improved and high capability single server we can save the energy and also give advantage of dynamic resource allocation to the cloud users. Proposed system is Design and implements to build single machine VM cluster to provide a web session management load balancer.

The rest of this paper is organized as follows. The Literature review is discussed in section 2. Section 3 explains problem definition with Research methodology and Section 4 describes the system architecture and proposed work. Section 5 includes module implementation and finally, section 6 concludes and discusses future scope of this work.

II. LITERATURE REVIEW

"Cloud computing is a new approach that reduces IT complexity by leveraging the efficient pooling of on-demand, self-managed virtual infrastructure, consumed as a service"[1]. According to Wikipedia "Cloud computing is Internet-based computing, whereby shared resource, software, and information are provided to computers and other devices on demand, like the electricity grid" [2].

Cloud Computing is defined by a large-scale distributed computing paradigm that is driven by economies of scale, in which a pool of abstracted, virtualized, dynamically-scalable, managed computing power, storage, platforms, and services are delivered on demand to external customers over the Internet[3].

Virtualization is a technique to implement cloud computing resources such as platform, application, storage, and network. The beauty of virtualization solutions is that you can run multiple virtual machines (VM) simultaneously on a single computer. In 2010, J. Ritinghouse and J. Ransome [5] had explained implementation details of creating virtual platform i.e. virtual guest operating system running on Windows7 host .[4,5] This gives details about the installation of hypervisor and virtual machines.

In 2013, authors Zhen Xiao, Weijia Song, and Qi Chen [6] propose a new algorithm, such that it follow skewness algorithm operating in parts i.e. load prediction, hot spot mitigation, and green computing. For load prediction, apply the FUSD algorithm to each VM and each PM. They evaluate the performance of different VM to PM ratios, to reduce load on server they used hot spot mitigation so to distribute residual resource to make it fully utilized in the future as similar to our approach.

In this system, for transparent migration of virtual infrastructures between client and server, involving many challenges such as IP address sharing, bandwidth sharing and isolation from local home network traffic etc[11]. The papers on Personal Cloud by Hari, T. Lakshman, R. Viswanathan, and "Transperant migration of VM infrastructure" by R. Bifulco, R. Canonico, G. Ventre, V. Manetti [7, 8] provide solutions for such challenges of managing a personal Cloud, and also propose and implement an optimal solution to the resource management problem, allowing peers to share VMs across their individual Personal Clouds by specifying their resource offers and requests, and verify its performance via detailed simulations.

III. RESEARCH METHODOLOGY

Load balancing is a computer networking method for distributing workloads across multiple computers or a computer cluster, network links, central processing units, disk drives, or other resources. Successful load balancing optimizes resource use, maximizes throughput, minimizes response time, and avoids overload. Using multiple components with load balancing instead of a single component may increase reliability. Load balancing is usually provided by dedicated software or hardware, such as a multilayer switch or a Domain Name System server Process.

Hardware or *platform virtualization* refers to the creation of a virtual machine that acts like a real computer with an operating system. Software executed on these virtual machines is separated from the underlying hardware resources. For example, a computer that is running Microsoft Windows may host a virtual machine that looks like a computer with the Ubuntu Linux operating system; Ubuntu-based software can be run on the virtual machine. In hardware virtualization, the host machine is the actual machine on which the virtualization takes place, and the guest machine is the virtual machine. The words host and guest are used to distinguish the software that runs on the physical machine from the software that runs on the virtual machine. The software or firmware that creates a virtual machine on the host hardware is called a hypervisor or Virtual Machine Manager.

Cloud computing is a colloquial expression used to describe a variety of different types of computing concepts that involve a large number of computers connected through a real-time communication network. [9] Cloud computing is jargon term without a commonly accepted unequivocal scientific or technical definition. In science, cloud computing is a synonym for distributed computing over a network and means the ability to run a program on many connected computers at the same time. The phrase is also, more commonly used to refer to network-based services which appear to be provided by real server hardware, which in fact are served up by virtual hardware, simulated by software running on one or more real machines. Such virtual servers do not physically exist and can therefore be moved around and scaled up (or down) on the fly without affecting the end user - arguably, rather like a cloud. The popularity of the term can be attributed to its use in marketing to sell hosted services in the sense of application service provisioning that run client server software on a remote location.

It is evident from the progress of our survey that none of the load balancing techniques are efficient in avoiding deadlocks among the VMs. Hence, a load balancing algorithm to avoid deadlock by incorporating the migration has been proposed. Figure 1, depicts the design of the cloud architecture for this approach.

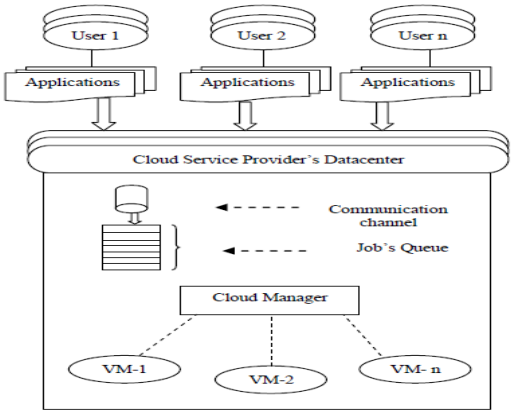

Figure 1: Cloud Architecture for load balancing

According to this design various users submit their diverse applications to the cloud service provider through a communication channel. The Cloud Manager in the cloud service provider's datacenter is the prime entity to distribute the execution load among all the VMs by keeping track of the status of the VM [6, 12, 13]. The Cloud Manager maintains a data structure containing the VM ID, Job ID of the jobs that has to be allocated to the corresponding VM and VM Status to keep track of the load distribution. The VM Status represents the percentage of utilization. The Cloud Manager allocates the resources and distributes the load as per the data structure. The Cloud Manager analyzes the VM status routinely to distribute the execution load evenly. In course of processing, if any VM is overloaded then the jobs are migrated to the VM which is underutilized by tracking the data structure. If there are more than one available VM then the assignment is based on the least hop time. On completion of the execution, the Cloud Manager automatically updates the data structure.

Load can be of any type but for simplicity image processing is considered a good candidate for the application of parallel processing because for processing the large volumes of data required more time degrading the performance. In this paper, we consider a task of classifying the large volume of data into three general categories such as red, green and blue for parallel processing.

Proposed Work:

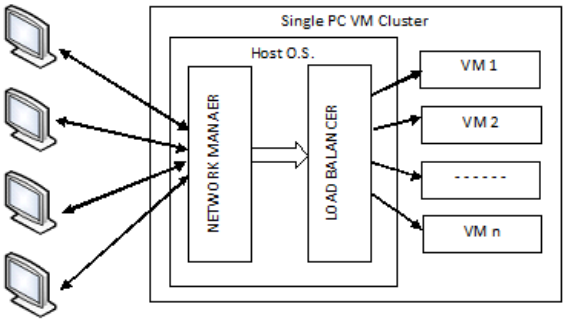

Figure 2: Proposed system architecture

Consider a scenario where several Host running in network, with some host running underutilized.

- We will first implement server client (master- slave) concept to let communicate multiple system with each other on priority level.
- Monitoring such underutilized machine status(idle machines), and make their resources available to Cloud
- Prepare a program for graphical monitoring and statistics analysis
- Prepare host program to monitor and to control web session load on the Cloud
- Prepare software for Host machine, to utilize connected client resources, such as task processing functionality to both server and client.
- Develop a random session generator program (to show web session load on host machine)
- Develop load distribution algorithm, in which Master PC will get the load and separate it in to equal number of client available and distribute the list to be processed by each client.
- Generate load balancing concept so that if any of the client goes offline and it left some work to complete then again distribution process will start by master PC to all clients.
- Each client node will report to the master about completion of each file. This will help server to keep track of work completed by each client.
- Test a setup

IV. ARCHITECTURE

The architecture shown in figure.1 is mainly comprised of three main phases.

- Resource Monitoring
- Socket Establishment
- Parallel computing

4.1 Resource Monitoring

In this phase, status of currently running machines of particular group (or organization) connected in network is being monitored. The status including memory usage and CPU utilization. Based on percentage of utilization of resources, access of resources is gets transferred to the cloud manager (master) in the form of connection request, by comparing it with suitable threshold value.

4.2 Socket Establishment

We use a simple novel approach for the construction of client server architecture. The communication between server and client occurs only after connection establishment. Server i.e. master is always in listen mode, to listen incoming connection request from client side. Server keeps track of each connected client along with its status in the form of records. Server also maintains the queue for processing the incoming task. This may be executed in parallel by several available client machines depending upon need for processing.

4.3 Parallel computing

In this phase master will distribute the task to all slave and they will perform their allocated task individually. In this way load balancing between all the machines will be maintained. If in case any client machine gets fail due to any reason? In that case load will again get balance by the remaining machines. So failure management will also be there.

*1) **Master / Slave System:***

Parallel processing system shown in figure 2 built using server / client technology. Where master server act as a process manager system. In proposed system whenever network initialize or the first system user started in the network will check for active server in the network if no server running found then it will become host or master server and other were become slave system. This feature can be dynamic or static which means user can disable or enable this feature.

*2) **Task (Image processing)***

This is the main input to the parallel processing system. First user need to define the task to perform. In proposed system it will consider a task of pixel color identification of image. Image can be passed to the client to process or it can be placed on shared data storage location.

*3) **Task Assigning***

Once the user provides input task on master server then master will analyse the task and divide it in to proper task and create its task list for client. Once all clients get the task list to process it will start processing. As proposed system will work on shared data storage it will reduce network processing and network traffic by removing data transfer.

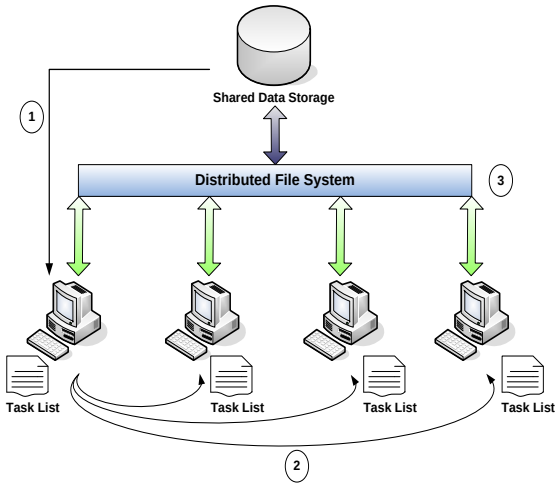

Figure 3: Parallel Computing

VM Load balancing Algorithm

Step 1. The Load Balancer maintains an index table of VMs and the number of requests currently allocated to the VM. At the start all VM's have 0 allocations.

Step 2. When a request to allocate a new VM from the Datacentre Controller arrives, it parses the table and identifies the least loaded VM. If there are more than one, the first identified is selected.

Step 3. The Load Balancer returns the VM ID to the Datacenter Controller.

Step 4. The Datacenter Controller sends the request to the VM identified by that id.

Step 5. Datacenter Controller notifies the Load Balancer of the new allocation.

Step 6. The Load Balancer updates the allocation table incrementing the allocations count for that VM.

Step 7. When the VM finishes processing the request, and the Datacenter Controller receives the response cloudlet, it notifies the Load Balancer of the VM de-allocation.

Step 8. The Load Balancer updates the allocation table by decrementing the allocation count for the VM by one.

Step 9. Continue from step 2.

In the existing algorithm there exists a communication between the Load Balancer and the Datacenter Controller for updating the index table leading to an overhead.

*4) **Parallel Processing:***

After master distributes the task over the distributed network all clients will process simultaneously and send acknowledgement to the master server. Master server will also process the task and at the same time will check for the client processing status and monitor it on the screen.

*5) **Process Failure Detection System:***

Proposed system will also manage failure of the client system at run time. Consider a condition if any client get failed due to any reason the remaining task should be processed by other system in the network. Master server will take care of this. It will continuously get acknowledgment from client time to time after each task completion. Once any client's connection get closed server will check work remain by specific client and then again divide this task and pass it to other client and client will process it.

*6) **Distributed File System:***

System sends data to client for processing but it will increase system overhead. So in proposed system data to processed will kept on shared storage and accessed using distributed file system so that network protocol processing can be reduced.

V. MODULE IMPLEMENTATION

A. Monitoring:

Status of running machines network is being monitored. Which including memory usage and CPU

utilization. Based on the threshold selected to connect the ideal client to the master is determined.

B. Task for processing:

- Task for proposed system is
- ❑ Image Processing or Wav files to MP3 format conversion
- ❑ Batch file Conversion
- ❑ Simultaneously on available number of client
- ❑ Each client will process same number of files
- Distribution of files to process
- Not distribution of part of files
- Using O.S. Provided API Functions

C. Network Connectivity:

- Server and client connected to each other using some logical link
- This link called as "SOCKET"
- Using Functions like
- ❑ Listen
- ❑ Connect
- ❑ Accept
- ❑ SendData, GetData

D. Process Distribution:

- User provides input task on master server
- Master will analyse the task and divide
- Create its task list for client
- Clients get the task list to process
- All work on shared data storage
- Reduce network processing and network traffic by removing data transfer processing.

E. Process Failure Management:

- Managing failure of the client system at run time.
- Continuously get acknowledgment from client time to time after each task completion.
- Once any client's connection get Status
- Check work remain by client
- Again divide this task and pass it to other client and client will process it.

VI. CONCLUSION AND FUTURE WORK

The expected outcome of the system is to save more energy and resources and providing cloud computing advantages to applications by establishing the single machine VM cluster and developing the load balancer for VM web server and all together should provide better web session management and dynamic resource allocation with minimum power utilization by controlling the VM state.

VII. REFERENCES

[1] http://www.vmware.com/ap/cloud-computing.html

[2] http://en.wikipedia.org/wiki/Cloud_computing

[3] Ian Foster, Yong Zhao, Ioan Raicu, Shiyong Lu, "Cloud Computing and Grid Computing 360-Degree Compared", Grid Computing Environments workshop, GCE'08, 12-16 Nov.2008

[4] http://www.virtualbox.org/wiki/Documentation

[5] John W. Ritinghouse and James F. Ransome Press, "Cloud Computing: Implementation Management and Security", ISBN:978-1-4398-0680-7,2010

[6] Zhen Xiao, Weijia Song, and Qi Chen, "Dynamic Resource Allocation using Virtual Machines for Cloud Computing Environment", IEEE June 2013 (vol. 24 no. 6) pp. 1107-1117

[7] Adiseshu Hari, T.V. Lakshman, Ramesh Viswanathan, Y. J. Chang, "The Personal Cloud — Design, Architecture and Matchmaking Algorithms for Resource Management", Hot-ICE'12 Proceedings of the 2nd USENIX conference, CA USA ©2012

[8] Roberto Bifulco, Roberto Canonico, Giorgio Ventre, Vittorio Manetti, "Transparent migration of virtual infrastructures in large datacenters for Cloud Computing", Italy 978-1-4577-0681-3/11©2011 IEEE

[9] http://en.wikipedia.org/wiki/cloud_computing#cite_note_Mariana_Carroll_2012-1

[10] Mladen A. Vouk , "Cloud Computing – Issues, Research and implementatio-ns",Dept. CS North Carolina State University, USA JCIT 16, 2008, 4, 235–246 doi:10.2498/cit.1001391

[11] C. Papagianni, A. Leivadeas, S. Papavassiliou, V. Maglaris, C. Cervell´o-Pastor and A. Monje, "On the optimal allocation of virtual resources in cloud computing networks",IEEE transactions on computers, revised on December 06, 2012.

[12] Manan D. Shah, Harshad B. Prajapati, "Reallocation and Allocation of Virtual Machines in Cloud Computing" Available :http://arxiv.org/ftp/arxiv/papers/1304/1304.3978.pdf