Dynamic Creation and Placement of Virtual Machine Using CloudSim

Vikash

Rao Pahalad Singh College of Engineering, Balana, India

Abstract—Cloud Computing becomes a new trend in computing. The IaaS(Infrastructure as a Service) is served by the cloud computing service providers by placing the virtual machines over the physical machines in a datacenter. The datacenter’s hardware get virtualized using virtualization component of cloud computing, before the placing of VMs on PMs, so that the resources of datacenter can be allocated to more than one VMs. A Virtual Machine placement problem is typically a search problem of available space and resources. Given that N is a set of virtual machines and M is a set of physical machines and the physical machines already hosting N1 virtual machines. Then the VM placement algorithms give a mapping of N+N1 machines on the M physical machines under a set of constraints. Resources in a datacenter can be provisioned to VMs by VM-scheduler in either a timeshared or spaceshared manner. After the successful placement of VMs on the PMs the tasks of the users are assigned to execute on different VMs. In this paper we are going to place the VMs and cloudlets dynamically and examine the resource allocation to VMs in a spaceshared and timeshared manner. Our simulation work is done using the cloud computing simulation toolkit known as CloudSim.

Keywords— Cloud Computing, VM Scheduler, Data Center, Host, Timeshared, Spaceshared, CloudSim, Cloudlets.

I. INTRODUCTION

As density of data is increasing day by day, due to which it need a high processing power and high cost resources. These resources are not affordable by the user. So, this requirement of users has led to the development of Cloud Computing technology in which high cost infrastructure resources are installed in a single datacenter and are accessed by the user through internet. A cloud computing service can give the following type of services to the end users: SaaS(Software as a Service), PaaS(Platform as a Service), IaaS(Infrastructure as a Service), and DSaaS(Data Storage as a Service). These services are used by users on pay-as-you-use manner.

In traditional days one server in datacenter can hold up only one operating system and all tasks are submitted to that particular server. Due to which the physical server get over utilized.

But due to cloud computing in these days a physical machine(host) in datacenter can have multiple operating systems at the same time. These multiple operating systems on a single host are the instances of operating system. These instances are referred here as virtual machines(VM). VM’s include virtual hardware and a real software package. The VMs gives an entire setting for application to run similar to they might on their own individual server[1]. The VMs are responsible for the handling of cloud users requests. In IaaS(Infrastructure as a Service) model requests for infrastructure are served by allocating VMs to those requests[2]. Therefore on service providers aspect the first concern is the allocation of these VMs on the particular physical resources i.e. the underlying hardware infrastructure. This allocation of VMs on physical machines ought to be economical in order that resources utilization is maximum and resource can be served in less time.

Virtualization in cloud computing allows a great deal of flexibility in the provisioning and placement of servers and their associated workloads in a data center. The problem of provisioning virtual machines is receiving a great deal of attention[4,5,6], while placement is often a relatively static manual process. A virtual machine using only virtual resources provides a computational encapsulation that may be moved between physical hosts in its entirety. A feature enabled by this encapsulation is the capability to migrate a running virtual machine from one physical host to another without a significant interruption of service[7]. This technology is currently available in VMware’s ESX product[8] and Xen[9], and it is anticipated in Microsoft’s Viridian product[10]. This document refers to this technology as live migration. We define a migration domain as a set of physical hosts among which a virtual machine can be migrated if sufficient resources are available.

The VM placement in cloud computing are often created by considering one of the three classes: reservation, on-demand, and spot market. In case of reservation user must pay a particular fee for a selected amount for every instance of a virtual machine.

In case of on-demand access user requests virtual machine for immediate access for a relatively short interval of time, and pay the charge for the used time. In case of spot markets the users ought to specify the price they're willing to buy the requested virtual machines, as in spot markets there's frequent fluctuation in providers value. The users are allotted with virtual machines only if the provider's value is same or less than the user's fee.

The hypervisor contains the list of all the physical resources available in the datacenters and information about their states. Since clients or users are provided with the computing resources only according to their requirement so the rest of it can be provided to other users. Many open source cloud computing solutions like Xen Cloud Platform, Nimbus, OpenNebula, Eucalyptus, TPlatform, Apache Virtual Computing lab, Enomaly etc. [13] have addressed this issue. But the framework of OpenNebula and Eucalyptus is quite impressive.

As multiple VMs can run on a single host at the same time, so, the large number of user’s requests are handled in a time and space efficient manner. The VMs are installed by means of hypervisors. A hypervisor is an operating system sits at the lowest level of the hardware environment. The instances of operating system i.e. the VMs are installed over hypervisor. A hypervisor provides a uniform abstraction of the underlying physical machine allowing multiple VMs to execute simultaneously.

The service level requirements such as availability, capability of resources, performance and costs of resources must be qualified by the provisioning of resources. All kind of users must be taken into care by resource provisioning and it must fulfill the requirements of all its end users and providers. So, a developer of resource provisioning technique must take care of the needs of Cloud Users on one hand and Cloud Provider on the other hand. Cloud users expect their jobs to be completed in fastest possible manner with high availability of resources and on the other hand the cloud provider invests huge capital with the aim to maximize the use of available resources in efficient and effective manner. Cloud users and providers both wants maximum throughput and improved performance for the money they invest.

II. RELATED WORK

Gupta A. et al. [16] proposed a HPC aware scheduler for Infrastructure as a Service cloud for improving performance of HPC applications in cloud. They have addressed the problem of adequate allocation of assorted range of application by using monotonous pool of resources. Service level agreement (SLA) is also taken into consideration.

R. Bajaj and D.P.Aggarwal in their paper[18] described that scheduling is a process of finding the efficient mapping of tasks to the suitable resources so that execution can be completed such as minimization of execution time as specified by customer. They also describe the various types of scheduling like static, dynamic, centralized, hierarchical, distributed, cooperative and non-cooperative scheduling. The scheduling problems in cloud and the types of users CCU(Cloud Computing Users) and CCSP(Cloud Computing Service Providers) are also described by them.

R. N. Calheiros, Rajiv Ranjan, Anton Beloglazov, C.A.F De Rose, Rajkumar Buyya [17] described about the Simulation techniques and the CloudSim. They described the various features of CloudSim like it supports for modelling and simulation for large scale of cloud computing infrastructure including data centers on a single physical computing node.

III. THE VIRTUAL MACHINE PLACEMENT APPROACHES

The capacity of physical host and the VM’s requirements must be well known for placing the VM on the physical host in a correct manner.
The goal of placement of virtual machine is either to save the consumption of power by shutting down some servers or to maximizing the usage of resources. Both the goals must be keep in mind while designing the autonomic VM placement algorithm. The VM placement algorithms are classified into two major categories:

1) Approach based on Quality of Service, and
2) Power based Approach.

The power management necessity is increasing in computing environments. The power management is needed due to: a) the increasing demands of power by computing. b) the consumption of power in cooling the resources in a data center. c) and due to the rising cost of power. The main aim of these approaches is the mapping of virtual machines to physical machines in such a way that the physical machine (PM) can be utilized to their maximum efficiency, and the other servers which are idle can be either hibernated or shut down depending on load conditions. The aim of the QOS based approach algorithms is to maximizing the quality of service (QOS) delivered by continuously monitoring virtual machine activity and employing advanced policies for dynamic workload placement. The QOS based algorithms are used for better utilization of resources and less frequent overload situations leading to savings in cost.

**VM placement Problem:**

A Virtual Machine placement problem is typically a search problem of available space and resources. Given that N is a set of virtual machines and M is a set of physical machines and the physical machines already hosting N1 virtual machines. Then the VM placement algorithms give a mapping of N+N1 machines on the M physical machines under a set of constraints. Resources in a datacenter can be provisioned to VMS in either a timeshared or space shared manner.

**IV. VM PLACEMENT ALGORITHMS**

**Rank Scheduling Algorithm:**

Open-nebula is an open source toolkit which uses match making scheduler as a VM scheduler, which uses a predefined rank[14][13] to prioritize the available resources. The VM having the highest priority is placed on a host[15]. The rank scheduling algorithm takes input as the requirements and the defined rank of VMs and the output is the resource number on which the VM is to be placed.

**Greedy First Fit Scheduling Algorithm:**

Eucalyptus and Nimbus uses Greedy-First-Fit-Algorithm[16] for the VM placement on a host. In greedy scheduling algorithm the node which can run a VM found first is selected as a host. The inputs to the greedy-first-fit algorithm are the VM request and requirement and the output is the resource number which is assigned to the requested VM.

**Round-Robin Scheduling Algorithm:**

Eucalyptus and Nimbus open-sources for cloud solution also use Round-Robin algorithm which, records the last position of the scheduler visited[16].

**V. TASK SCHEDULING IN CLOUD COMPUTING**

Tasks in IaaS model in cloud computing are the requested resources which are completed by means of VMs. In IaaS when a cloud user requests for resources, then these requests are accepted by the broker which then uses the task scheduling to assign the tasks to different VMs. These VMs are then placed over the physical hosts which are scheduled by the VM scheduler. The scheduler at the broker level will decide the provisioning of resources to the VMs. More than one tasks can be assigned to a single VM. These tasks are then scheduled by the task scheduled in a time-shared or space-shared environment.

**Time-Shared Policy:**

In a time-shared environment task scheduler will distribute the capacity of core among different tasks dynamically.

**Space-Shared Policy:**

In a space-shared environment the scheduler assigns specific CPU cores to specific task.

**VI. EXPERIMENTAL SETUPS**

The simulation is used Window7 Ultimate(64-bit), dual-core processor, 2.16GHz with 3GB memory. For the simulation we need special toolkit known as CloudSim toolkit. CloudSim is a library for simulation of cloud computing scenarios. The simulation of time-shared and space-shared policies is simulated using CloudSim. The simulation includes the execution of cloudlets by VMs and the allocation of these VMs on physical hosts.
Simulation Life Cycle:

a) Datacenter in CloudSim is created by:
void getDataCenter()

{String inputa;
int noOfDataCenter=0;
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
try
{System.out.println("Enter the no of data centers:");
inputa=br.readLine();
noOfDataCenter=Integer.parseInt(inputa);}
catch(Exception e){}
for(int i=0;i<noOfDataCenter;i++)
{Log.printLine("DataCenter-"+i+"is creating");
Datacenter datacenter = createDatacenter("Datacenter_"+i);
Log.printLine("DataCenter-"+i+"is created");}
}

b) RAM for VM can be assigned to a VM at run-time by:

public void setRam(int ram)

{String inputa;
BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
try{System.out.println("Enter the ram");
inputa = br.readLine();
ram=Integer.parseInt(inputa);}
catch(Exception e){}
this.ram = ram;}

Table 1

<table>
<thead>
<tr>
<th>Host ID</th>
<th>Pes</th>
<th>RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2</td>
<td>2048</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>4096</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>8192</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>4046</td>
</tr>
</tbody>
</table>

And we requested 8 VMs to be placed on physical host. After simulation completed we observe the allocation of resources to VMs by the VM schedulers.

Table 2

<table>
<thead>
<tr>
<th>VM</th>
<th>Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Host-1</td>
</tr>
<tr>
<td>1</td>
<td>Host-2</td>
</tr>
<tr>
<td>2</td>
<td>Host-1</td>
</tr>
<tr>
<td>3</td>
<td>Host-2</td>
</tr>
<tr>
<td>4</td>
<td>Host-0</td>
</tr>
<tr>
<td>5</td>
<td>Host-1</td>
</tr>
<tr>
<td>6</td>
<td>Host-2</td>
</tr>
<tr>
<td>7</td>
<td>Host-3</td>
</tr>
</tbody>
</table>

After allocation of VMs to the physical hosts 13 cloudlets are being executed on these VMs. And the sequence of execution of these cloudlets is being observed in cloudlet timeshared and spaceshared scheduling policy.
Table 3: Sequence of Execution of Cloudlet

<table>
<thead>
<tr>
<th>Time-Shared Cloudlet Scheduler</th>
<th>Space-Shared Cloudlet Scheduler</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cloudlet</td>
<td>VM</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>12</td>
<td>4</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>11</td>
<td>3</td>
</tr>
</tbody>
</table>

After completing cloudlet execution VMs are get destroyed in a on-demand access.

VII. CONCLUSION

VMs are placed on physical hosts and are capable to execute the cloudlets. These VMs are scheduled on physical hosts in a timeshared and spaceshared policies. After placement of VMs on hosts cloudlets are send to VMs to execute. These cloudlets are also scheduled by the cloudlet scheduler in a timeshared and spaceshared manner. After completion of cloudlets execution VMs are destroyed from the datacenter. In simulation using CloudSim toolkit we can request a VM at the run time.

REFERENCES


