Task Scheduling in virtual machines

Enroll. No.	: 10503869
Name of Student	: Rishi Bhardwaj
Name of supervisor(s)	: Mrs. Sangeeta Mittal

DEC – 2014

Submitted in partial fulfillment of the Degree of

5 Year Dual Degree Programme B. Tech - M. Tech

In

Computer Science Engineering

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING & INFORMATION TECHNOLOGY

JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY, NOIDA

(I)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

List of symbols and acronyms

S. No	Symbols and Acronyms
1	**VM**- virtual machine
2	**API**- Application Programming Interface
3	**CPU**- Central Processing Unit
4	**VMM**- Virtual Machine Monitor
5	**PV**- Paravirtualization
6	**OS**- Operating System
7	**Dom0**- domain 0
8	**DomU**- Domain Unprivileged

1. ABSTRACT

Cloud computing is a new emerging trend in computer technology that has influenced every other entity in the entire industry, whether it is in the public sector or private sector with the advance feature of the cloud there are new possibility opening up how application can be built and how different services can be offered to the end user through Virtualization, on the internet. Considering the growing importance of cloud finding new way to improve cloud services is an area of concern and research focus. In available Virtual Machine Load Balancing policies limitation of cloud is that they don't save the state of the previous allocation of virtual machine to a request from the user and the VM Load Balancing algorithm require execution each time a new request for VM allocation received from user. This problem can be resolve by developing an efficient VM load balancing algorithm. Currently load balancing algorithms like round robin, weighted round robin are not efficient in some scenario when there is large number of virtual machine. Our objective is to develop an effective load balancing algorithm using Virtual Machine Monitoring to maximize or minimize different performance parameters(throughput for example, no. of interaction per virtual machine) for the Clouds of different sizes (virtual topology de-pending on the application requirement). Virtual Machine enables the abstraction of an OS and Application running on it from the hardware. Allocating the request that comes to server is done using algorithm so that our resource utilization will be maximum. These requests are application elements sandboxed within VMs based on the decision made by the load balancer. We can also call this task scheduling in virtual machine because we are scheduling the task of server to virtual machine. We can use simulator or real time system to implement the load balancing in cloud computing. Some simulators are like cloudsim, cloudanalyst provide full simulation environment. Some real time system are like Xen, VMware provide live environment. I am using XEN as hypervisor to implement the load balancing algorithm. Application Load Balancer on Xen is a very important tool in today's world as we are completing transferring our work on cloud platforms so scheduling of task properly on various servers is very important. This tool helps the client as the application requested by client is directly transferred to the most suitable server according to its need. So by use of this the client does not need to manually select the server.

2. Results of revised Literature Survey

2.1 Revised problem statement

Task scheduling is a technique used to schedule task on virtual machines. Task scheduling aims to optimize resource use, maximize throughput, minimize response time, and avoid overload of any one of the resources. Application Load Balancer on Xen is a very important tool in today's world as we are completing transferring our work on cloud platforms so scheduling of task properly on various servers is very important. This tool helps the client as the application requested by client is directly transferred to the most suitable server according to its need. So by use of this the client does not need to manually select the server. In general we generally allocate the work to that machine that is next in queue. For this I have to dynamically manage our workload on all virtual machines to check weather which machine have lowest workload. And as a complete check on load, memory of various servers is maintained from time to time so have the complete status of all the servers all the time. By this we are able to utilize the resources completely and efficiently. Thus, the throughput and response time is improved efficiently. Task scheduling technique can be used in many datacenters because in datacenters we have data and application that need to be computed so with the scheduling the task we can balance the load of virtual machine without manually. With our application client not need to select manually that in which VM our client application should run, application will be automatically run on VM according to CPU utilization.

2.2 Revised solution approach to revised problem in terms of technology /platform to be used.

XEN:-

The Xen Project hypervisor is an open-source type-1 or baremetal hypervisor, which makes it possible to run many instances of an operating system or indeed different operating systems in parallel on a single machine (or host). The Xen Project hypervisor is the only type-1 hypervisor that is available as open source. It is used as the basis for a number of different commercial and open source applications, such as: server virtualization, Infrastructure as a Service (IaaS), desktop virtualization, security applications, embedded and hardware appliances. The Xen Project hypervisor is powering the largest clouds in production today.

Here are some of the Xen Project hypervisor's key features:

- Small footprint and interface (is around 1MB in size). Because it uses a microkernel design, with a small memory footprint and limited interface to the guest, it is more robust and secure than other hypervisors.
- Operating system agnostic: Most installations run with Linux as the main control stack (aka "domain 0"). But a number of other operating systems can be used instead, including NetBSD and OpenSolaris.
- Paravirtualization: Fully paravirtualized guests have been optimized to run as a virtual machine. This allows the guests to run much faster than with hardware extensions (HVM). Additionally, the hypervisor can run on hardware that doesn't support virtualization extensions.

Python

Python is a widely used general-purpose, high-level programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than would be possible in languages such as C. The language provides constructs intended to enable clear programs on both a small and large scale. Python supports multiple programming paradigms, including object-oriented, imperative and functional programming or procedural styles. It features a dynamic type system and automatic memory management and has a large standard library.

Tabular comparison between tools and software:-

Tool	**Benefit/Advantages**	**Drawbacks/limitation**
Xen	• Open Source • Mostly used • Type 1 hypervisor • Real system	• Tough to configure • No GUI

CloudSim	• Easy to setup • User friendly	• Simulator based
Virtual Box	• Easy setup with GUI	• No privilege
Citrix	• Hardware Assisted Virtualization • Paravirtualization	• Less privilege

Table II

2.3. Revised Tabular comparison of other existing approaches/ solution to the problem framed

No.	**Research Paper**	**Issue /Challenges**	**Implementation / Solution**	**Tool used**
1	Benefits and Challenges of Three Cloud Computing Service Models	Proposed issue related to IaaS, PaaS, and SaaS.	• Selection of cloud • Security	None
2	An Advanced Survey on Cloud Computing and State-of-the-art Research Issues	Architectural Design of Data Centre Distributed File System over Cloud Distributed Application Framework over Clouds	• VMM Migration • Server consolidation • Information Security	None
3	A Load Balancing Algorithm with Key Resource Relevance for Virtual Cluster	Load balancing algorithms	Proposed hybrid solution group based load balancing algorithm	None

4	Round Robin Approach for VM Load Balancing Algorithm in Cloud Computing Environment	Algorithm issue in cloud environment	State based round robin algorithms	Cloudsim
5	On Load Balancing for Distributed Multiagent Computing	Load balancing approach	Agent based load balancing algorithm	None
6	A Model Based Task scheduling Method in IaaS Cloud	hard to predict the requirement of virtual machine based on the workloads	model that forecast the load and estimate the resource requirement	Xen
7	A Scheduling Strategy on Load Balancing of Virtual Machine Resources in Cloud Computing Environment	Seldom considers system variation and historical data for load balancing and scheduling	Genetic algorithm for load balancing	Platform ISF OpenNebula
8	Predictive VM consolidation on multiple resources: Beyond load balancing	Unpredictable system and workload delays	• Fair load balancing scheme • Queueing network analytic model	Xen virtualization testbed
9	Using CloudSim to Model and Simulate Cloud Computing Environment		Use of cloudsim to avoid configuration of real virtualization environment	cloudsim

Table II

2.4 Overall Conclusion of Literature Review

Cloud computing is a way of leveraging the Internet to consume software or other IT services on demand. Users share processing power, storage space, bandwidth, memory, and software. With cloud computing, the resources are shared and so are the costs. Users can pay as they go and only use what they need at any given time, keeping cost to the user down.

Load balancing is a technique used to schedule task on virtual machines. Load balancing aims to optimize resource use, maximize throughput, minimize response time, and avoid overload of any one of the resources. Using multiple components with load balancing instead of a single component may increase reliability through redundancy. Load balancing is usually provided by dedicated software or hardware, such as a multilayer switch or a Domain Name System server process.

Xen runs in a more privileged CPU state than any other software on the machine. Responsibilities of the hypervisor include memory management and CPU scheduling of all virtual machines ("domains"), and for launching the most privileged domain ("dom0") - the only virtual machine which by default has direct access to hardware. From the dom0 the hypervisor can be managed and unprivileged domains ("DomU") can be launched.

The dom0 domain is typically a version of Linux, or BSD. User domains may either be traditional operating systems, such as Microsoft Windows under which privileged instructions are provided by hardware virtualization instructions (if the host processor supports x86 virtualization, e.g., Intel VT-x and AMD-V), or *para-virtualized* operating system whereby the operating system is aware that it is running inside a virtual machine, and so makes hypercalls directly, rather than issuing privileged instructions. Xen boots from a bootloader such as GNU GRUB, and then usually loads a paravirtualized host operating system into the host domain (dom0).

3. Improvised Analysis, Design and Modeling

3.1 Functional requirements and Non Functional requirements

Functional Requirements

- Multiple guest OS can run on host OS via Xen virtual machine monitor.
- The resource usage of any application to be calculated so as to profile the applications based on their resource usage.
- Live migration between virtual machines so as to migrate VM in case of any failure or allocation of an application to a specific VM.
- Different characteristics to be given to different virtual machines and then the applications to be run on these VMs accordingly.
- View all running VMs and hosts and their live performance & resource utilization statistics.
- see performance & utilization statistics for each VM

Non Functional Requirements

.

- Response time- This requirement describes how much time it takes from the moment a user sends a request to the system, until a complete response is provided. In web applications, this com- prehends request transmission and processing, and response transmission. The factors that account for it are resource capabilities —processing power, memory, disk, network latency and bandwidth— and the load produced by other processes running in the server or the number of concurrent requests. For complex requests, this may also involve calls to external systems, or to other subsystems, in which case the host's internal network characteristics and other resources' load may be taken into account.
- Uptime- The total time the service is available. It may be expressed as a percentage. When considering this requirement, it is necessary to take into account the provider's own uptime. For example, if a provider has an uptime of 99.5%, it would be impossible to deploy an application with a higher uptime. Other factors involve the recoverability of the system (i.e., how much time it takes to restart the service after a failure happens).

- Requests per unit of time- This requirement describes the number of requests the system can handle successfully per unit of time, and can also be referred to as the system's throughput. Resource allocation and usage has an impact in this parameter. Additionally, the number of requests can have an impact in the response time requirement (i.e., a high number of requests will result in a deterioration of the overall response time). Fault tolerance. One of the system's properties is how it can withstand errors, either hardware or software-based. In the case of cloud, non-software errors can be generated either at the physical or the virtual machines hosting the service.
- Security- Security is another requirement that can be applied to the cloud provider or to the developed system.

3.2 Use case and Sequence diagrams of proposed solution

Use case diagram for XEN configuration and virtual machine installation

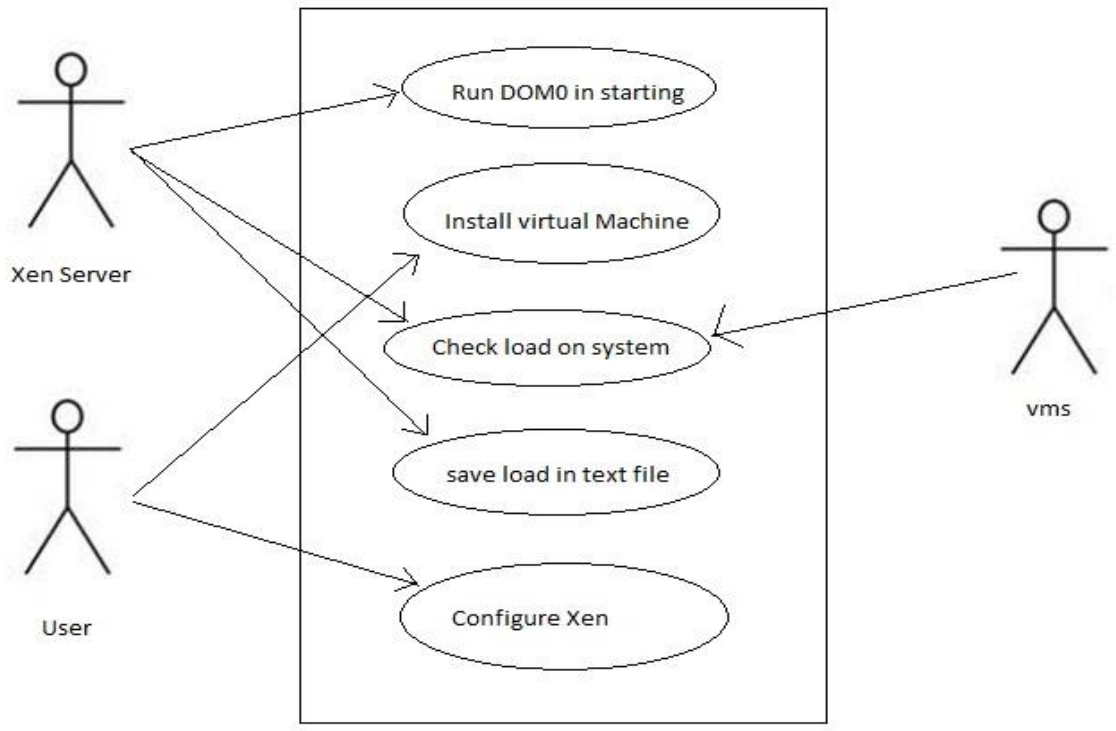


Fig. 5 Use case diagram for XEN configuration

- XEN server start at the time of booting Ubuntu it starts the Dom0 which is a preconfigured virtual machine and automatically start at the time of boot.
- For installing virtual machine on server user can use VMM.
- Check load on system function can be used to check the complete load on system in server.
- Save load to txt file function can be used to save the load in text file.

Use case diagram for scheduling

This diagram shows the all the functions that will our task scheduler will do.

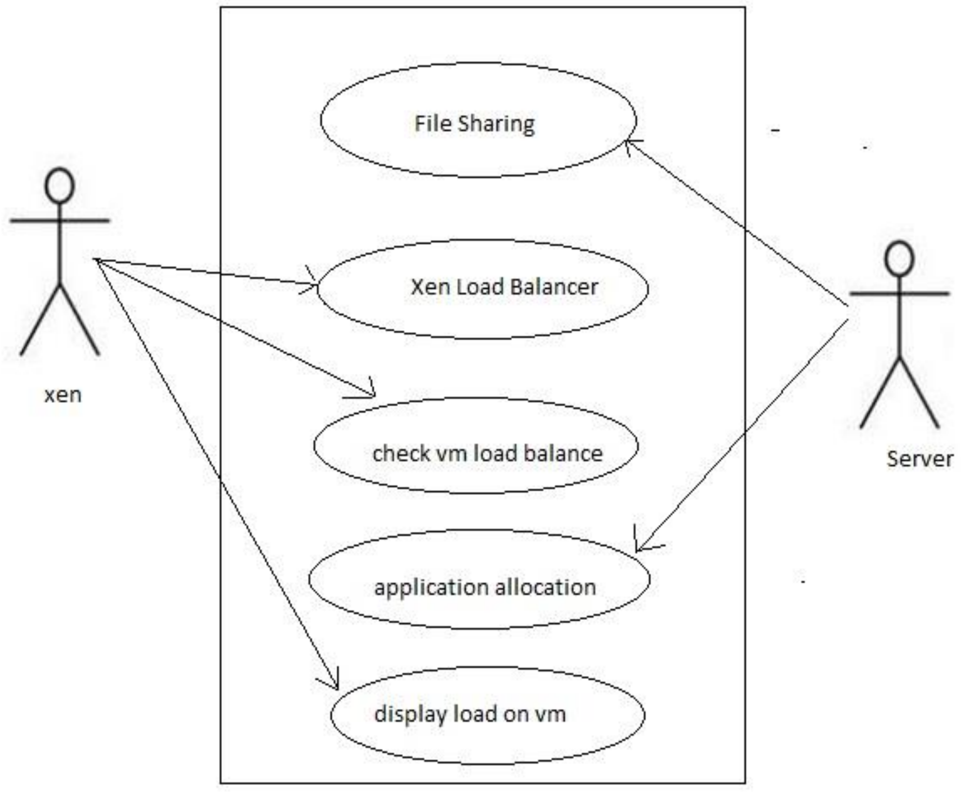


Fig. 6 Use case diagram for task scheduling

- File sharing will be used to create a network supported shared file system in this all the VMs will be able to access a common space.
- Xen task scheduler will be used to balance the load between the virtual machines.
- Application allocation function will allocate the desired application to that VM which has less resources

Sequence Diagram:-

Sequence diagram for load balancing

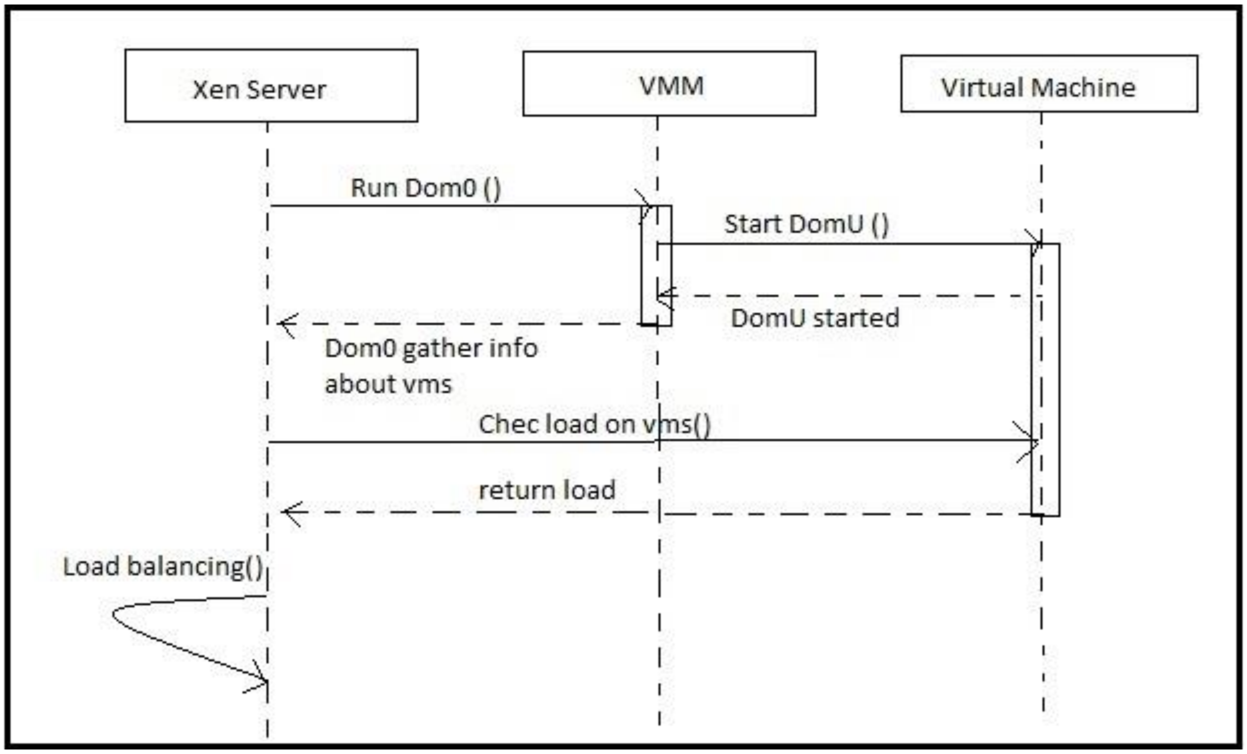


Fig. 7 sequence diagram

3.3 <u>Algorithms</u>

<u>Static Scheduling</u>

This scheduling algorithm that distributes the load based strictly on a fixed set of rules relating to the characteristics of input load. It does not consider which node is receiving more or less load. In all static algorithms final selection of the virtual machine is done as soon as the application is created and cannot be altered during execution to change the load of that virtual machine. These static load balancing techniques are suitable for a system in which load is limited and request of the clients is also limited. But nowadays load on cloud servers is also increasing and also the load is not static hence we need more efficient algorithms then static load balancing algorithms.

Round Robin Algorithms

Whenever a new application comes it is assigned a virtual machine in a round robin fashion. In general, basic idea for Round Robin is to reduce message passing between various virtual machines and reduce communication delay. Thus it is independent of the state of the system. When coming applications are of similar load then Round Robin works very well as it reduces the communication delay due to inter-process communication. Thus Round Robin has best performance for this special purpose application of similar load, but does not give a good performance for general cases.

I have implemented round robin algorithm and count the total number of interaction that are made to the VMs to allocate the application to server.

No. of virtual machine	No. of Application	No. of times algorithm runs	Average time
1	1	1	1
2	3	7	2.33
2	5	11	2.20
3	7	9	1.28

Weighted round robin algorithms

Weighted round robin algorithm is the improvised version of **round robin algorithm.** In weighted round robin algorithm we have assigned the weighted to every virtual machine so the number of request that a virtual machine can handle is depended on the weight of virtual machine and the load on virtual machine.

Weight of virtual machine 1 = 2

Weight of virtual machine 2 = 1

Weight of virtual machine 3 =2

If no. of virtual machine are two that means we are considering the top 2 virtual machine and others are on sleep mode.

No. of virtual machine	No. of Application	No. of times algorithm runs	Average time
1	1	1	1
2	3	4	1.33
2	5	7	1.60
3	7	8	1.14

So we can see the complexity reduce when we use the weighted round robin algorithm.

Calculation of load in virtual machine:-

Xen provide the facility to calculate the load on CPU. Each virtual machine has some no. of VCPU so if we want to find the how much load a VM has we have to use this formula.

$$\text{Load on virtual machine} = \frac{\text{CPU(\%)}}{\text{No.of VCPU}}$$

If CPU (%) for VM1 is 150% and no of virtual CPU are 2 then load on VM1 will be

$$= 150/2=75$$

Hybrid algorithm

Group based load balancing algorithm

Define the group of each virtual machine. A group can have one or more than one virtual machine in it.

As the resources in the virtual cluster system are normally heterogeneous $[R_r]$, = (R_{r1}, Rr2.......R_{rn}) is adopted to represent the virtual resource vector in virtual resource group. 1 and is the number of virtual resource in the virtual resource group. To calculate the integrated load $L(r_{rj})$of virtual resource r_{rj} in the virtual resource group r_r, the major concern lies in CPU utilization rate $R_{cpu(rrj)}$, memory utilization rate $R_{mem(rrj)}$, occupancy rate of network band width $R_{net(rrj)}$ and disk utilization rate $R_{disk(rrj)}$ of each virtual resource. The calculation formulation calculate the integrated load of virtual resource r_{rj} in the virtual resource group r_r is shown as follows:

$L\ (r_{rj}) = k_1 * R_{(rrj)} + k_2 * R_{mem(rrj)} + k_3 * R_{net(rrj)} + k_4 * R_{disk(rrj)}$

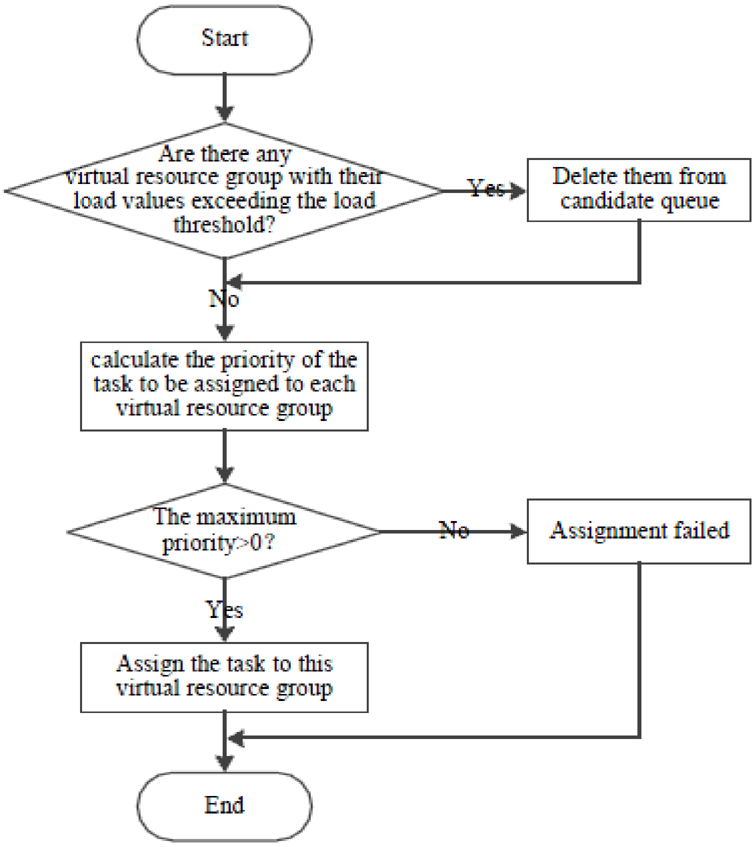
$j=1, 2\ldots nr, \Sigma k=1$

Where: k_d, d=1, 2, 3, 4 is the weight parameter of each index, reflecting the influence of different kinds of indexes on the load value, configured by users. Now we can calculate the average value of all the virtual resources.

Inter-resource group priority scheduling strategies

Input: A series of random tasks $\{t_1, t_2\ldots t_m\}$, a set of virtual resource groups $\{r_1, r_2\ldots r_n\}$;

```
Output: Each task has been assigned to a virtual resource group;
For i=1 to m // m is the number of tasks;
{
 Update the load value of every virtual resource group;
Delete the virtual resource groups of which load values exceed the threshold value from candidate queue;
For j=1 to n // n is the number of virtual resource groups;
 Calculate the priority of task $t_i$ to be assigned to virtual resource group rj;
If the max priority value > 0 Assign task $t_i$ to virtual resource group which has max priority;
Else Prompt user the task $t_i$ has failed to be assigned;
 }
```


Start
Are there any virtual resource group with their load values exceeding the load threshold?
Yes
Delete them from candidate queue
No
calculate the priority of the task to be assigned to each virtual resource group
The maximum priority>0?
No
Assignment failed
Yes
Assign the task to this virtual resource group
End

Priority scheduling process

Load balancing scheduling strategies within a resource groups

After the priority scheduler assigns the task to one certain kind of virtual resource group, the load balancing scheduler of this virtual resource group will then assign this task to one of its virtual resources. Because the relevance of this task to each virtual resource in this one same virtual resource group is mutually identical, the difference only lies in the hardware performance.

Input: A series of random tasks {t_1, t_2… t_k}, a group of virtual resources;

Output: Each task has been assigned to a appropriate virtual resource;

Var: Succeed_Flag : boolean init : FALSE // mark whether the task has been assigned successfully or not; Wait_Flag : boolean init : FALSE // mark if the task is needed to be added to waiting queue or not;

```
For i=1 To k // k is the number of tasks;
 {
```

```
Update the load value of each virtual resource;
Divide the virtual resources into five grades {$g_1$, $g_2$, $g_3$, $g_4$, $g_5$} from the minimum to the maximum load value;
Succeed_Flag=Wait_Flag=FALSE;
 For j=1 To 5
{
 If the weight value of $t_i$ corresponds to grade $g_j$
{
 If there are some virtual resources in grade $g_j$
{Assign $t_i$ to the virtual resource which has minimum load value on grade $g_j$;
Succeed_Flag=TRUE;
 Break;
}
Else
{ If there is an appropriate virtual resource on grade $g_{i+1}$ to $g_5${
 Assign $t_i$ to this virtual resource;
Succeed_Flag=TRUE;
Break;
 }Else
{
Wait_Flag=TRUE;
Break;
 } }}
// end If
 }
// end For If !Succeed_Flag && Wait_Flag Add $t_i$ to the waiting queue;
Else If! Succeed_Flag &&! Wait_Flag Prompt user the weight value of $t_i$ is wrong, assignment failed;
}
```

4. Implementation

4.1 Overall implementation details and issues related to it.

XEN Server Configuration:-

For project propose I have configured the Xen hypervisor in Ubuntu 12.04 LTS version. Xen is a native (bare-metal) hypervisor providing services that allow multiple computer operating systems to execute on the same computer hardware concurrently.

Below are the steps for configuring the Xen hypervisor in Ubuntu 12.04.

Install Xen Package

```
$ sudo apt-cache search xen
```

```
$ sudo apt-get install xen-hypervisor-4.1-amd64 xen-utils-4.1 xen-tools xen-docs-4.1
```

Install libvirt and Virtual Manager Tools

```
$ sudo apt-cache search virt-install
```

```
$ sudo apt-get install virtinst python-libvirt virt-viewer virt-manager
```

Create Xen Virtual Machine Images

```
$ sudo dd if=/dev/zero of=/etc/xen/vm01.img bs=1M count=3072

3072+0 records in

3072+0 records out

3221225472 bytes (3.2 GB) copied, 39.5561 s, 81.4 MB/s
```

View Dom0 Current Status

```
$ sudo xm list

Name                                    ID   Mem VCPUs      State   Time(s)

Domain-0                                  0   945     2     r-----     73.1


$ sudo mv /etc/grub.d/10_linux /etc/grub.d/50_linux

$ sudo update-grub2
$ sudo reboot
```

Modify the Configuration File

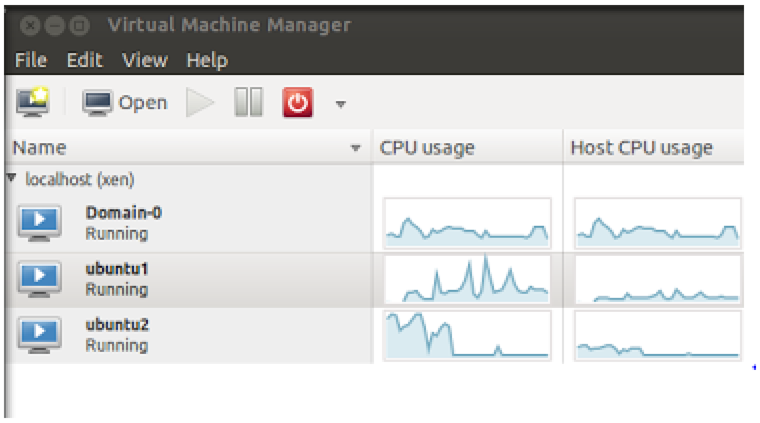
```
$ sudo vi /etc/xen/xend-config.sxp
```

Modify #(xend-unix-server no) to (xend-unix-server yes), save the file and then restart the xend service

```
$ sudo xend restart
```

<u>Virtual machine monitor</u>

A virtual machine monitor (VMM) is a host program that allows a single computer to support multiple, identical execution environments. All the users see their systems as self-contained computers isolated from other users, even though every user is served by the same machine. In this context, a virtual machine is an operating system (OS) that is managed by an underlying control program.



Name	CPU usage	Host CPU usage
localhost (xen)		
Domain-0 Running		
ubuntu1 Running		
ubuntu2 Running		

Figure 8

Dynamically load calculation:-

I have implemented a script for dynamically calculating the load on virtual machine so that we can use the updated load on each virtual machine. This scripts run in every 10s and update the load on text file.

NFS Configuration:-

NFS mounts work to share a directory between several virtual servers. This has the advantage of saving disk space, as the home directory is only kept on one virtual private server, and others can connect to it over the network. When setting up mounts, NFS is most effective for permanent fixtures that should always be accessible.

Setting Up the NFS Server

Start off by using apt-get to install the nfs programs.

```
apt-get install nfs-kernel-server portmap
```

```
nano /etc/exports
```

Add the following lines to the bottom of the file, sharing both directories with the client:

```
/home           12.33.44.555(rw,sync,no_root_squash,no_subtree_check)
/var/nfs        12.33.44.555(rw,sync,no_subtree_check)
```

Once you have entered in the settings for each directory, run the following command to export them:

```
exportfs –a
```

Setting Up the NFS Client

```
apt-get install nfs-common portmap
```

Once the programs have been downloaded to the the client server, create the directories that will contain the NFS shared files

```
mkdir -p /mnt/nfs/home
mkdir -p /mnt/nfs/var/nfs
```

Then go ahead and mount them

```
mount 12.34.56.789:/home /mnt/nfs/home
mount 12.34.56.789:/var/nfs /mnt/nfs/var/nfs
```

Testing the NFS Mount

Once you have successfully mounted your NFS directories, you can test that they work by creating files on the Client and checking their availability on the Server.

Create a file in each directory to try it out:

```
touch /mnt/nfs/home/example /mnt/nfs/var/nfs/example
```

You should then be able to find the files on the Server in the /home and /var/nfs directories.

```
ls /home

ls /var/nfs/
```

Implementation of round robin algorithm

I have done implementation of round robin algorithm in python to assign the task to virtual machines. With the use of text file generated of loads of VMs the algorithm is checking that which virtual machine is suitable to allocate the task. And it tells server to that which virtual machine to tell user to works on.

Implementation of weighted round robin algorithm

Weighted round robin is same as round robin but in case of WRR we have allocated the weighted to the each virtual machine. I have assigned the weigh according to number of VCPU to the VMs. Now each VM will handle the request according to their weight.

Related Issue:-

- Currently I have installed two virtual machines in Xen. Xen cause much overheating to laptop and may be shut down any time.
- Scalability issue is also there in Xen, I have installed two VMs now but it is not possible to check the algorithms without scalable platform. At least 5 VMs must be there to efficiently evaluate the algorithm.
- Not able to open the application from server to host.

4.2 <u>Test Plan and Test Cases</u>

Test Plan

Type Of Test	**Test Performed?**	**Comments**	**Software Component**
Unit Testing	Yes	The application is mainly divided into the main components – Xen Hypervisor, Virtual machines, migrating environment, profilation.	Profilation patch, hypervisor, virtual machine after running different operating systems on them.
Integration Testing	Yes	The application's modules are inter- Dependent. 'Dom u' host depends upon 'Dom0' host And eventually Dom0 depends on the underlying hardware	Integration of hypervisor with all the virtual machines, profilation patch and global manager.
Performance Testing	Yes	All the virtual machines should be test separately and performance on correspondent domains should checked.	Performance of every virtual machine should be done.
Stress testing	No	At this stage, we have not identified stress points for this application as feature development is higher priority.	
Security Testing	Yes	As hypervisor is the underlying managing body under virtual machines. Hence any hacker	Several components were tested due to high risk involved.

		can enter hypervisor and hence can cause huge damage to the system	
Load Testing	Yes	Load Testing is important at this stage As we are dealing with a variety of virtual machines and multiple operating systems hence load testing in very important.	Underling hardware was tested along with load on virtual machines.

Table V

Test Case:-

Test Id	Input	Expected Output	Result
1	Virtual machines with different configuration	New guest OS gets created using their allocated resources	Pass
2	Script run for dynamic updating the load	Update all load data in text file	Pass
3	Run the script of NFS	All virtual machine must able to see the shared files	Pass
4	Input load for check which VMs is accurate	Name of the VMs suitable for allocation	Pass

5	Open application to defined VM through server	An application must be open in guest OS automatically	Fail
6	Implement algorithm of nth number of Vms	Algorithm must run	Pass in some cases

Appendix

A. Project Plan as Gantt chart or WBS

Timeline	Work
10 Aug to 20 Aug	Research on load balancing algorithm
21 Aug to 10 Sep	Xen Server configuration
10 Sep to 20 sep	Virtual machine installation
1 Oct to 10 Oct	Deciding the parameter of load balancing and value calculation
11 Oct to 20 Oct	NFS server configuration
21 Oct to 30 Oct	Project report and remaining work
1 Nov to 4 Nov	Project report
5 Nov to 10 Nov	Dynamic updating of the load
21 Nov to 30 Nov	Round robin algorithm implementation
1 Dec to 10 Dec	Weighted round robin algorithm implementation
27 Dec to 3 Jan	Report and error remove

B. References

[1] Joel Gibson, Darren Eveleigh, Robin Rondeau, Qing Tan, “Benefits and Challenges of Three Cloud Computing Service Models”, 2012 Fourth International Conference on Computational Aspects of Social Networks.

[2] Mohiuddin Ahmed1, Abu Sina Md. Raju Chowdhury, Mustaq Ahmed, Md. Mahmudul Hasan Rafee, "An Advanced Survey on Cloud Computing and State-of-the-art Research Issues*",* International Journal of Computer Science Issues.

[3] Michael Hauck, Matthias Huber, Markus Klems, *"*Challenges and Opportunities of Cloud Computing*",* Karlsruhe Institute of Technology Technical Report Vol. 2010-19.

[4] Wang Xiaojing, Tong Wei, Zhao Wei, Liu Jingning, *"*Evaluation on network load balancing in Xen*", 2012*.

[5] Maha Jebalia, Asma Ben letaifa, Sami tabbane, "Live migration in virtual network environment", 2010, IEEE.

[6] Raghavendra Achar, Santhi Thilagam, Nihal Soans, P. V. Vikyath, "Load Balancing in Cloud Based on Live Migration of Virtual Machines", 2013, 2013 Annual IEEE India Conference.

[7] Jinhua Hu, Jianhua Gu, Guofei Sun, Tianhai Zhao, "A Scheduling Strategy on Load Balancing of Virtual Machine Resources in Cloud Computing Environment", 2010, "3rd International Symposium on Parallel Architectures, Algorithms and Programming".

[8] Lei Lu, Hui Zhang, Smirni, E, Guofei Jiang, Yoshihira, K., "Predictive VM consolidation on multiple resources: Beyond load balancing", 2013, Quality of Service (IWQoS), 2013 IEEE/ACM 21st International Symposium.

[9] Abels,T. , Dhawan,P. , and Chandrasekaran,B , "An Overview of Xen Virtualization", 2005, "Dell Document on virtualization".

[10] Wang Long, Lan Yuqing and Xia Qingxin," Using CloudSim to Model and Simulate Cloud Computing Environment", 2013, 2013 Ninth International Conference on Computational Intelligence and Security.