JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Workload Based Performance Enhancement in Cloud System Using VMM

Payal Panchbhayye
Assistant Professor
Departmnet of Artificial Intelligence and Data Science
Modern College of Engineering ,Pune ,India

Abstract : Cloud computing a form of grid computing deals with providing everything as a service. Cloud computing is mainly used in business and IT industry which supports heavy outsourcing model computational resource, where service availability, security and quality are essential features. The phenomenon of software aging, one in which the state of the software system degrades with time, has been reported. To counteract software aging, a technique called software rejuvenation has been proposed, which essentially involves occasionally terminating an application or a system, cleaning its internal state and/or its environment, and restarting it. Since rejuvenation incurs an overhead, an important research issue is to determine optimal times to initiate this action. In this work, I propose a technique to enhance the performance of VM under variable workload conditions using various timer policies.

IndexTerms **- Phase type distribution; Cloud computing; Time based rejuvenation; Dynamic availability**

I. INTRODUCTION

CLOUD computing is the wildest developing field in ICT. It completes computing demand progressively and supply services modelled after public utilities, like energy and gas usually available at high fees, at much lower costs, and according to a pay per use pricing schema [1]. The flexible behaviour of cloud systems support an increase of resource availability and fit's the requirements of users and their applications, and reducing costs. Cloud computing also allows to businesses a heavy outsourcing model for computational resources, essential features of which are service availability, security, and quality guarantees all to understand these problems is one of the factors that are going to steer the market in the coming years.

In particular, service availability is given primary importance being one of the main requirements and affects user satisfaction. Research has been done in recent years to find the optimal infrastructure size and configuration allowing guaranteeing the desired availability level.Performance of software often gets lower when it is used for long time [1]. As we use software for a longer time period it shows some error like degradation of software, crash or hangs failure, memory leaking, memory fragmentation, unreleased file locks, data corruption, storage space fragmentation and round of error. This phenomenon of decrease in software performance is called as Software aging [1] [2].

Software rejuvenation [6] is a technique which deals with the software fault and it is cost effective technique. This technique manages the software fault by cleaning the system internal state and prevents the occurrence of more faults. Software aging and rejuvenation has two separate approaches first is measurement based and second is analytical. Measurement based approach is focus on statistically analysis of measured data on resource availability and also predict the expected time to resource enervation

Cloud computing is a structure which consist of shared hardware and software resources and embedded on the basis of virtualization [8] .Virtualization allows to keep multiple virtual machines (VMs) on top of a physical machine and which is managed by virtual machine monitor (VMM). Attention is focus on the possibility of applying rejuvenation to moderate the effect of software aging in the VMM because software degradation mainly influences on long-term running applications and services. Continuous phase type (CPH) distributions are used to characterize the VMM time to failure. Expanded process is used to model the system availability that allows keeping memory of the age reached by them. Ms changes according to the conservation of reliability principle when hosted. Kronecker algebra is used to represent the expanded process through symbols .Workload-dependent system behaviour is allowed to be formally represented by this in a way that it is to be implemented in a software tool.

We also focus on time-based rejuvenation model this model is used to find optimal rejuvenation timer .This timer minimize objective function. In this timer policy we are setting timer at the time of system start but it is same in whole process it does not change as workload going to increase in large process.

RELATED WORK

Author in the paper [1] focus software rejuvenation in cloud system using variable timer policy in case of heavy workload
Software rejuvenation technique allows preventing the occurrence of software failures which is explained by author [6]. Two main rejuvenation approaches can be model-based and measurement based.
Time-based rejuvenation is explained by Jie Li, and Dengyi Zhang [7] who gives optimal timer to system by which rejuvenation is performed. Jie Li, and Dengyi Zhang also showed that the rate at which software ages is not constant .This rate is depends on the workload which varies according to time.

Aye MyatMyatPaing and Ni LarThein in the paper [6][14] propose a comprehensive model to study the software rejuvenation of a UNIX operating system which also considers system workload the analytical and measurement-based techniques are combined by them which first collects and analyze data to study the influence of the workload.

AutranMacêdo and Taís B. Ferreira in the paper [4][15] explores the concept of cloud computing, its advantages and disadvantages and describes several existing cloud computing platforms and discuss the results of quantitative experiments carried out using PlanetLab, a cloud computing platform as well. A two-phase scheduling algorithm under a three-level cloud computing network is a scheduling algorithm combines OLB (Opportunistic Load Balancing) and LBMM (Load Balance Min-Min) scheduling algorithms that can utilize more better executing efficiency and maintain the load balancing of system [6][8][9].

Autran Macêdo and Fumio Machida in paper [5] [10] have focus on memory related software aging issues which causes aging related failures. They have mainly focused on memory leak problems. They have discussed important drawbacks of using well known system-wide and application-specific aging indicators, as well as propose effectual solutions for both cases. Memory-related aging effects are caused by memory leak and memory fragmentation problems [11] [12]. Memory leak is a software defect that is mainly caused by incorrect use of memory management routines [13] [14]. A memory leak occurs when an application process dynamically allocates memory blocks and, for some reason, does not release them back to the OS during its runtime. Here the authors have tried to find out memory leaks in a system both in user level and kernel level by the use of aging indicators.

Domenico Cotroneo & Roberto Natella, in paper [3] have focus on virtualized w2qenvironment for software aging defect. In this paper aging occurrence is detected by conducting experiments on physical and virtual machines and identifies the differences between the two, and proposes a feature code-based methodology for failure prediction through system call, then execute a prototype in virtual machine manager layer to predict failure time and rejuvenate transparently. They have conducted four experiments to detect the aging phenomenon in physical machines (PM) and virtual machines, and then calculate the aging rate for comparison. For the experiment they have collected three kinds of system resource as metrics that signal software aging Memory resources such as Free Memory, Active Memory ,CPU resources including User and System Time, IO resources such as Block Read\Write Count, IO Waiting Time. During stastical analysis they have found decreasing of free memory size.

The authors in paper [6] have presented the issues of performability management in a virtualized data center (VDC) that hosts multiple services using virtualization. Performability is a concept of a mixed metric of performance and availability. The users of a VDC generally request a certain level of application performance in a service level agreement (SLA). VDC providers need to decide an optimal server configuration and management operations for guaranteeing application performance and maximizing the availability. They have focused on placement algorithm of VMs and rejuvenation schedules for VMs and VMM in a VDC.

The authors in paper [4] also have proposed a work where they combined server virtualization technique that rejuvenates both VMs and VMMs. To maximize the resource utilization they have used live VM migration for shifting of VMs when VMMs are rejuvenated. By the help of simulation technique they have showed the high availability of VMs with maximum resource utilization in virtualized data center.

PROPOSED SYSTEM AND DESIGN

3.1 Problem Statement

The proposed statement of the system is to propose a new innovative approach to model software aging in cloud system and also in LAN network. Hence, I propose a technique to model and evaluate the VMM aging process and to investigate the optimal rejuvenation policy that maximizes the VMM availability under variable workload conditions. Starting from dynamic reliability theory and adopting symbolic algebraic techniques, we investigate and compare existing time-based VMM rejuvenation policies. I also propose a time-based policy that adapts the rejuvenation timer to the VMM workload condition improving the system availability.

3.2 System Design

Proposed system is compatible in case of heavy workload. There are many technique which deals with workload but there are some draw back for existing technique .In existing technique like software rejuvenation if workload occurs system gets restart or shut down but it is not useful in case of large application as there may be chances of memory leak and memory fragmentation both in physical machines and virtual machines. I am going to propose a technique in which work will distribute automatically on different virtual machine connected in server to avoid workload .For distributing this workload I am using fixed timer policy. In this Policy the timer is set at system start-up and it does not change with respect to the system dynamics (e.g., system workload variations).

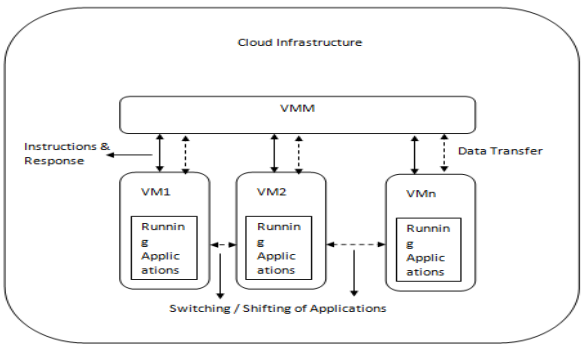


Fig 1. System architecture

Application consists of mainly two modules/parts which are again divided into sub modules.
These main two main modules are VMM and Node

VMM is a virtual machine monitor, having higher level of physical machine. But in actual implementation, VMM is a virtual machine residing on a physical machine. Hence our application can consist of a VMM application located on physical machine, for which, here after, called as VMM Manager or only as Manager.

The Node is nothing but a machine which is under monitoring by VMM Manager. Paper stated that it might be workstation or any other device connected in network. But as paper deals with only software failures, it must be workstation such that a physical machine connected in network. Hence, nodes are physical machines, might be slaves or clients connected in network, either internet or intranet (LAN), and Manager becomes their server.

3.3 Plan of execution

Node: The node application at client side, such that the machine which is under monitoring by VMM manager, continually perform the following tasks

1. Read Task Manager of machine, for understanding, number of applications running at given Time, memory used by them, and status of applications such that whether they are running or Get hanged.
2. Check memory space available on each drive, to check whether that Drive is running out of Memory.
3. Try to distinguish between network applications, such that the Applications which are used by network users and local applications which are accessed only on local Machine.
4. Send all above collected information to VMM manager for monitoring.
5. Check for any instructions from VMM manager and follow them.
6. All these tasks are performed at regular intervals set by VMM.

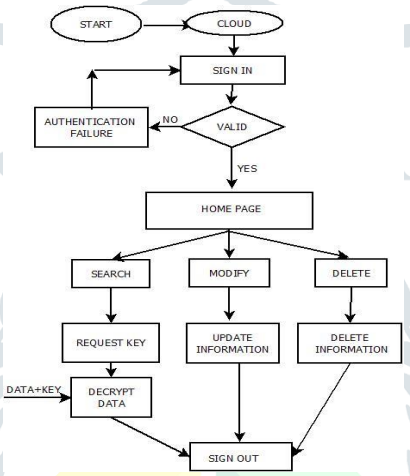


Fig 2. System flow diagram

3.4 Proposed Algorithm

The proposed algorithm steps are as below:

While implementing proposed algorithm, it would be assumed that all data was stored at central server.

1. Get details of Virtual Machine A (VM[A])
2. Get total running applications on VM[A]
3. Count total memory occupied by all running applications on VM[A] and its status
4. If status is running then jump to step 5 and if status is not responding jump to step 10
5. Consider/Assume threshold memory
6. If memory occupied on VM[A] is greater than threshold memory, then find out the application occupying max memory
7. Initiate LOOK method to find any other VM, called as VM[B] in same network having (total memory occupied + memory of considered application on VM[A]) less than its threshold memory
8. If found, send 'application close' command to VM[A] and 'application start' command to VM [B] (shift application from VM[A] to VM[B])
9. If not found, send instruction 'application close' on VM[A]
10. If application was not responding, send instruction 'application close' on VM[A] to forcefully close the application
11. Repeat steps 1 to 8 after pre- defined period

3.5 Mathematical Module

A) Module on Rejuvenation

X=Cnt(t_f)-Cnt(t_i) = Difference between running application count at two time instants

X = Difference of application count at two time instants

If X is zero, then check for acknowledgement from machine, if it gives, then exit, if not instruct to restart

RES= (X! =0)? (EXIT): (POST ACK) = If application count is same, check for ack message from machine, otherwise exit

(RES==1)? (RESTART): (EXIT)

B) Module on Workload Distribution

Y= $\sum_{k=0}^{cnt} \mathrm{Mem}(\mathrm{App}[k])$ = Total memory consumption, Mem_{th} = Threshold memory = min free memory required to run machine efficiently

RES=(Y>Mem_{th}) ? (Exit) : (A)

A=((Mem(App[k])>Mem(App[K+1))?Mem(App[K]):Mem(App[k+1])

Machine1: STOP (A) ⇔ Machine2: START (A) = Shift the application from machine 1 to machine

IV.WORK COMPLETED

4.1 Hardware and Software required

A) Input

Number of running application in the system

B) Hardware and Software used

Hardware Interfaces:

Processor- Core 2 Duo (Min)

Speed - 1.5 GHz (Min)

RAM - 2 GB (min)

Hard Disk - 200 GB

Floppy Drive - 1.44 M

Software Structure: (SERVER SIDE)

Operating system -Windows 2007

Coding Language -ASP.Net with C#

Data Base - SQL Server 200**8**

4.2 Practical work

Following figure shows practical work completed.

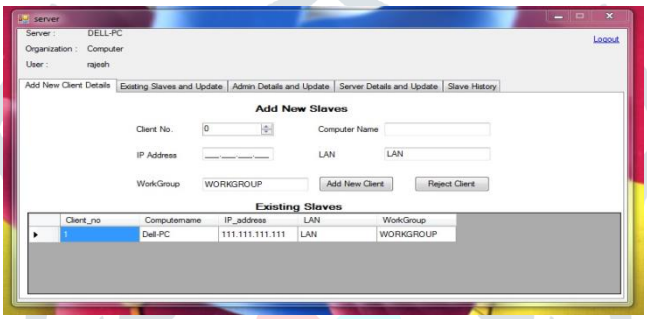


Fig 3. Running applications with their name and id

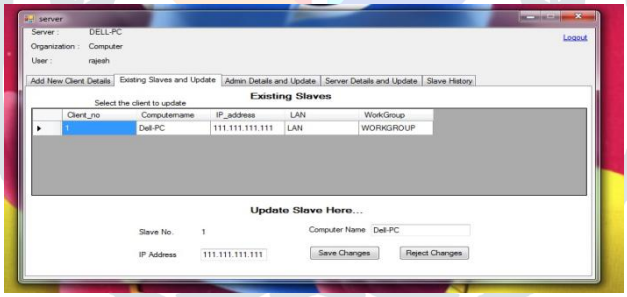


Fig 4. GUI response of application

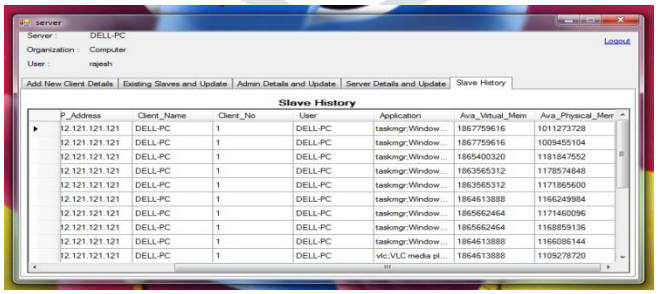


Fig 5. Available virtual & Physical memory on slave

4.3 Expected Result

Expected result can be as in following graph which gives steady state system availability at specific time interval

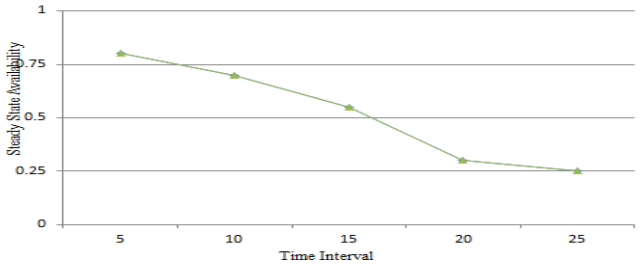
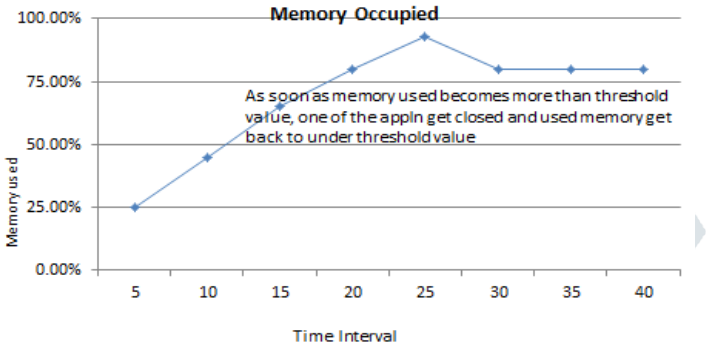


Fig 6 .Graph for System availability

Fig 7.Memory Occupation for system



V. CONCLUSION AND FUTURE SCOPE

In this paper I have discussed how performance of the system can be increased in case of heavy workload using VMM .I also deal with software rejuvenation, a specific form of environment diversity that is gaining importance as an effective preventive maintenance technique. The main contribution of my work is a measurement-based model that integrates the effect of system workload on operating system resources and an approach to investigate its effect on software aging. Since many studies have suggested strong correlations between workload and system reliability/ availability, this model is an improvement over the purely time-based model. I have also distinguished relation between the system workload and resource exhaustion.

I am trying to extend proposed technique in 1) Along with wired and wireless LAN i.e. in intranet, in future; I am going to try to implement this application in internet environment. 2) Implementation of application in internet i.e. in global network does not restrict by any boundaries and hence it was becomes possible to monitor and take care of any machine from anywhere.3) Also, along with solution to software failures, hardware failures detection and their solution are to be included in future.

REFERENCES

[1] Workload-Based Software Rejuvenation in Cloud Systems “IEEE Transaction on Computer 2013”

[2] Michael Grottke, RivalinoMatias Jr., and Kishor S. Trivedi, “The Fundamentals of Software Aging," IEEE, 2008

[3] DomenicoCotroneo, Roberto Natella, , Roberto Pietrantuono, and Stefano Russo, " A Survey of Software Aging and Rejuvenation Studies," ACM, Vol 5

[4] "Software Aging and Rejuvenation," Wiley & Sons, 2008

[5] AutranMacêdo, Taís B. Ferreira, and RivalinoMatiasJr, "The Mechanics of Memory-Related Software Aging," IEEE, 2011

[6] DomenicoCotroneo, and Roberto Natella, “Monitoring of Aging Software Systems affected by Integer Overflows," IEEE, 2012

[7] Kehua Su, Hongbo Fu, Jie Li, and Dengyi Zhang, "Software Rejuvenation in Virtualization Environment," IEEE, 2011

[8] JyotiprakashSahoo, SubasishMohapatra and, Radha Lath, "Virtualization: A Survey On Concepts, Taxonomy and Associated Security Issues," IEEE, 2010

[9] Kenichi Kourai, and Shigeru Chiba, "Fast Software Rejuvenation of Virtual Machine Monitors," IEEE, Vol 8, No 6, 2011

[10] Fumio Machida, Dong Seong Kim, Jong Sou Park, and Kishor S. Trivedi, “Toward Optimal Virtual Machine Placement and Rejuvenation Scheduling in a Virtualized Data Center," IEEE, 2008

[11] Kenichi Kourai, and Shigeru Chiba, "A Fast Rejuvenation Technique for Server Consolidation with Virtual Machines, " IEEE, 2007

[12] Fumio Machida, Jianwen Xiang, Kumiko Tadano, and Yoshiharu Maeno, "Combined Server Rejuvenation in a Virtualized Data Center”

[13] Thandar THEIN, Sung-Do CHI, and Jong Sou PARK," Availability Analysis and Improvement of Software Rejuvenation Using Virtualization," Economics and Applied Informatics, Years XIII, 2007