Volume 4, Issue 1, January 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Private Cloud Scheduling with SJF, Bound Waiting, Priority and Load Balancing

Ashish Kumar Singh[1], Sandeep Sahu[2]
Dept. of CS, SRIT, Jabalpur, MP, India

Kamalendra Kumar Gautam[3], Mangal Nath Tiwari[4]
Dept. of CS, APSU, Rewa, MP, India

Abstract: Cloud computing is an emerging field and prefer by many organization for to manage the group of computer system's hardware and software but it performance is lot more depend on the effective scheduling algorithm and load balancing. In this paper we address this issue and propose an algorithm for private cloud which has high throughput. To improve the throughput in private cloud SJF is used for scheduling and to overcome form the problem of starvation we use bounded waiting. For load balancing we monitor the load and dispatch the job to the least loaded VM. In private cloud there we consider that there are mainly two group of user one more important who are involve with controlling and important function of organization and other who are mainly involve with the daily routine task. In our algorithm we maintain two different queues for these two types of user. This will insure that the job of important employee or user is executed as early as possible.

Keywords— Private cloud, virtualization, load balancing, bounded waiting, VM and distributed Server.

I. INTRODUCTION

Cloud computing is the use of computing resources (hardware and software) that are delivered as a service over a network (typically the Internet). Load balancing and provisioning in cloud computing systems is really a challenge now. Always a distributed solution is required. Because it is not always practically feasible or cost efficient to maintain one or more idle services just as to fulfils the required demands Jobs cannot be assigned to appropriate servers and clients individually for efficient load balancing as cloud is a very complex structure and components are present throughout a wide spread area. Here some uncertainty is attached while jobs are assigned. The aim is to provide an evaluation and comparative study of these approaches.

Cloud Components

A Cloud system consists of 3 major components such as clients, data centre, and distributed servers. Each element has a definite purpose and plays a specific role.

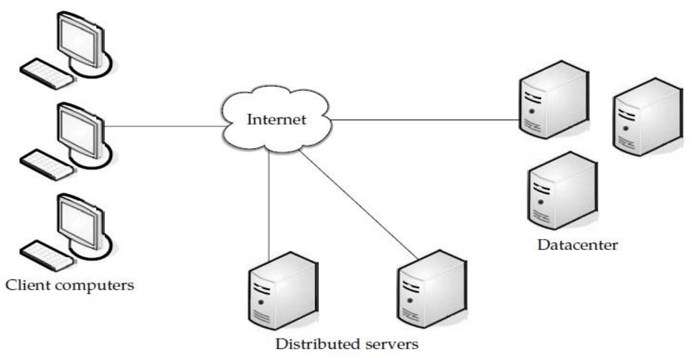

Figure1: components of cloud computing

Clients

End users interact with the clients to manage information related to the cloud Client generally fall into three categories as given in [15]:

- Mobile: Windows Mobile Smartphone, smart phones, like a Blackberry, orange, iPhone.
- Thin: They don't do any computation work. They only display the information.
- Servers do all the works for them. Thin clients don't have any internal memory.
- Thick: These use different browsers like IE or Mozilla Firefox or Google Chrome to connect to the Internet cloud.
- Now-a-days thin clients are more popular as compared to other clients because of their low price, security, low consumption of power, less noise, easily replaceable and repairable etc.

Datacenter

- Datacenter is nothing but a collection of servers hosting different applications. An end user connects to the datacenter to subscribe different applications. A datacenter may exist at a large distance from the clients.
- Now-a-days a concept called virtualization is used to install software that allows multiple instances of virtual server applications.

Distributed Servers

Distributed servers are the parts of a cloud which are present throughout the Internet hosting different applications. But while using the application from the cloud, the user will feel that he is using this application from its own machine.

Applications & types

Cloud computing has following applications or it provide following types of services

- Infrastructure as a service (IaaS)
- Platform as a service (PaaS)
- Software as a service (SaaS)
- Storage as a service (STaaS)
- Security as a service (SECaaS)
- Data as a service (DaaS)
- Test environment as a service (TEaaS)
- Desktop as a service (DaaS)
- API as a service (APIaaS)

Virtualization

It is a very useful concept in context of cloud systems. Virtualization means “something which isn’t real”, but gives all the facilities of a real. It is the software implementation of a computer which will execute different programs like a real machine. Virtualization is related to cloud, because using virtualization an end user can use different services of a cloud. The remote datacenter will provide different services in a full or partial virtualized manner.

Motivation

1. The use of cloud computing are increasing day by day because now people the use of smart phone and tablet PC are becoming common . People want same computing capability as they got on their stationary Desktop PC are Laptop computer. But these are small and have less computing facility so they can’t run heavy software. Cloud computing are used here to run the desired software on the mobile as a service over the Internet
2. The needs of organization are very dynamic today and if they go to purchase all things like infrastructure, software and platform they have to spend a lot of money every time and also spend time for the setup. They have to do a lot of documentation for maintaining the list of vendor, license key etc. With cloud environment organizations have to pay for the resource they want to use and they have to pay for according to approximately their usage time. They not have to spend almost any time for the setup.
3. With cloud computing organization have to worry about the maintenance because it is done by the cloud provider. So businessman can think more about their business.

RGPV portal and MPOnine Portal, IRCTC all are the example of cloud computing which shows how the work become easier for both manager and customer with the use of cloud computing But Cloud Computing performance have depend a lot more on the scheduling algorithm and proper load balancing algorithm. Scheduling algorithm will make such a sequence of process that throughputs are increased and load balancing algorithm divide the load properly between all available resources. Cloud Computing is a parallel processing model where these issue is of vital importance so they also have importance. As cloud is a pay-go-model, the business performance needs to be accelerated which is a challenging issue in the domain. There is certain important factor like

- Job Scheduling
- Load Balancing
- Resource Allocation

So we have chosen this topic to find a better strategy to improve the performance in cloud computing environment.

Paper Organization

In section two we review the previous work of various author of same field. We discuss about their work and also discuss about their limitations. In section three we propose our algorithm. We also evaluate the performance of algorithm by calculating their time complexity. In section four we conclude our work and discuss about the future work.

II. REVIEW OF PRIOR WORKS

Author of [1] has study the factor that influence the job rejection in the cloud environment. He shows the comparison of SJF and R-R scheduling algorithm in peak hour means when no of arrival of job is very high. Author suggests R-R scheduling for scheduling and SJF for load balancing and process migration to avoid deadlocks. Author shows in the result that in peak hour SJF has better performance in job rejection issue. SJF has the problem of starvation so long job has very high turn around time. His study shows that in the case of cloud computing the number of job rejection should be less because cloud is pay-go model so if customer’s job is rejected then the feature of cloud computing would not attract the customer .Author actually shows a comparisons between SJF and RR scheduling but he not suggest any way of changing the algorithm from SJF to RR and form RR to SJF author only says about the migration of processes for load balancing but how migration will occur and what is the method of knowing overloaded VM. Author has study the factor that influence the job rejection in cloud environment and he suggest the R-R algorithm for scheduling and SJF scheduling for load balancing but R-R scheduling has a lot of overhead of pre-emption and SJS has starvation problem.

Authors of [2] study a existing load balancing algorithm and enhance it for the load balancing. Author suggests that for the load balancing the cloud manager not have to take action only at the allocation time. Cloud manager periodically monitor the all VM's load and if cloud manager found that some VM are over loaded then cloud manager migrate the load form over-loaded VM to under utilized VM. In the proposed algorithm the Cloud Manager analyses the availability of the VMs at the time of job arrivals to update the data structure thereby having less overhead involved in maintenance of the data structure compared to the existing approach.

His algorithm is good for load balancing but his algorithm do the allocation of VM to a newly arrived client process only with the purpose of load balancing he not study other performance criteria for the performance improvement of cloud computing . Load balancing is not desired at all time it mainly desired at peak hour and on periodic base to balance the load equally on the VM.

Author of [6] propose a solution for managing large image collections. Author presents a cloud computing service and its application for the storage and analysis of very –large image. His solution allows that an input image can be divided into different sub-images that can be stored and processed separately by different agents in the system, facilitating processing very-large images in a parallel manner. His purpose is to create a cloud computing service capable of storing and analysing very- large image datasets. Author develop cloud computing service prototype for storing and analysing images. Adequate parallelism and workload balancing of our distributed system is crucial feature to ensure an improved performance. Author actually study real life application of very large image datasets and parallelism will really improves the performance such application. His solution divide large image into sub-images and improve the performance by parallelism. Author not covers issue of environment and cost and also of load balancing.

Author of [7] study on the issue which will affect the performance of cloud computing because in the case of cloud computing performance is the key requirement because customer want full return of his money . The presence of some dominant organization like Google , Amazon , Sales force , Azure make the field of cloud computing more fascinating so it is important for all the vendor that they improve the performance to be competent in the cloud market. Overall the cloud demonstrates four primary characteristics which attract the customer to prefer the cloud computing as compare to other computing:

a) Flexibility (Elasticity) and the capability of range (Scale) up and down.
b) Self – service provisioning and automated de-provisioning.
c) Program – development interface (APIs)
d) Charging and metering of assistance utilization in the form of a pay-as-you-go model.

He also identified that scheduling if key issue for the better performance of cloud system. Author studied performance issues in cloud computing and he found that scheduling algorithm is key issue for performance in cloud computing but he not suggest any solution for the better performance of cloud environment.

Author of [8] suggest the algebraic scheduling of the processes because he found that some different processes have different need for the execution so he suggest the algorithm which show the process resource demand in the form of utility function . He suggest that desired resource demand should be in the form soft constraint means it is not necessary that process can execute with the desired resource. As cloud resources and applications grow more heterogeneous, allocating the right resources to different tenants' activities increasingly depends upon understanding tradeoffs regarding their individual behaviours. One may require a specific amount of RAM, another may benefit from a GPU, and a third may benefit from executing on the same rack as a fourth.

Author modify the existing approach where resource consumers has to specify zero or more hard constraints with each request, based on some predetermined attribute schema understood by the cluster scheduler . Such constraints could serve as a filter on the set of machines, enabling identification of the subset that is suitable for the corresponding request. But, this approach ignores an important issue: in many cases, the desired machine characteristics provide benefit but are not mandatory. For example, running a new task on the same machine as another with which it communicates frequently can improve performance, but failing to do so does not prevent the task's execution—it just makes it somewhat less efficient. We refer to such cases as Soft constraints. Treating them as hard constraints can lead to unnecessary resource under-utilization, and ignoring them can lead to lower efficiency, making them an important consideration for cloud schedulers.

This paper proposes a specific approach for accommodating soft constraints, as well as hard constraints and general machine heterogeneity. In this model, each job submitted for processing is accompanied by a resource request, which is expressed as utility functions in the form of algebraic expressions indicating what benefit would be realized if particular resources were assigned to it. Author suggests algebraic scheduling because of heterogeneity in cloud environment. He suggests that process has to express their resource requirement in soft constraint and algebraic scheduling decide that process has to be executed on which VM. In his scheduling utility function express all option in sequence which will have a lot of overhead.

III. PROPOSED ALGORITHM FOR PRIVATE CLOUD

Important point of scheduling in private cloud

a) Jobs are admitted only of the network so there are less number of jobs compares to public cloud so some overhead permissible in scheduling.

b) Some user are at controlling level and some are at worker level there may be more user level but at least two level must be managed to ensure that the important task is executed as early as possible.
c) Higher throughput is primary requirement because if this is not the case then it is beneficial to switch to higher hardware and software cost means not to use cloud computing.
d) SJF scheduling has the higher throughput so this is better option of scheduling in private cloud
e) With SJF some jobs may suffer due large burst time this is called starvation. To overcome form this problem bounded waiting must be there which introduce a tag with every arrived and decreased automatically with every new arrival of process and when it will reach to zero then that job must be executed.
f) In cloud computing there is number of processor is available for the execution so a proper load balancing is also mandatory for this there should be a common queue for all jobs arrived form anywhere and form any user and cloud manager can allocate the job to the idle VM (Virtual Machine)
g) On summarizing we can say that
    i. We maintain two queue one for important task means the task coming form controller user and one for less important task. In every process PCB a field is introduced which either have the value of 1 or 0 depending upon form which system it will be generated . The value 1 indicate that process is coming from system reserved for important user and 0 indicate that the process is coming from system used by ordinary user .
    ii. On arrival every process priority is checked and it is put on the desired queue. These queues are maintained by the cloud manager which is running on the server machine which has the control over all resources. From a pool of processor any processor can become the server means a concept of distributed server is used which a common concept in case of cloud is computing so no problem of bottleneck is occurred. Server first assigns processors to the process of queue which has the 1 priority value. The 0 priority value queue's processes are executed only when the 1 priority value queue becomes empty.
    iii. Cloud manager check for the idle VM and any VM send a signal that it has finished the execution of current allocated job then cloud manger has to select a process for execution.
    iv. Cloud manage first check that is there is any job with zero tag which is assigned to the job on its arrival for ensuring bounded waiting if there is any process then that will be selected and dispatched to the least loaded VM .
    v. If there is no process with zero tag then cloud manager select the process with least burst time and allocate that job to the VM.

Private Cloud Algorithm
Database of VM is maintained at cloud manager which have 4 fields.
a) VM id
b) Num field indicate the number of process it can process.
c) p_cur Number of processe currently allocated to the VM.
d) load_per indicate load percent on the VM.

On PCB an extra tag field is used for the bound waiting and one for the priority value. Two Ready queues are maintained by cloud manager which contain the list of all arrived jobs of controller and worker user. We named them as Imp_Ready and W_Ready. Number of process is indicating by p_num1 in Imp_Ready and p_num2 in W_Ready.

```
Step – I → Initially all VM have the zero p_cur value in the database of VM and all n VM are idle
Step – II → While [W_Ready]!= NULL
Step – III → While [Imp_Ready]!=NULL
Step – III → if tag[process] ==0 then step –V else step – IV
Step – IV → Select min burst[process] form ready queue
Step – V → Select min load_per[VM] (least loaded and if more than one then least hop Time .
VM capacity indicated by the num field which must be more than the request size) VM from the VM pool and
p_cur[VM] = p_cur[VM] +1;
load_per[VM] = p_cur[VM]*100/num[VM]
Step – VI → remove the selected process form queue and dispatch it to selected VM p_num= p_num -1;
Step – VII → if new process is arrived add to the tail of the queue and p_num1= p_num1 + 1;
Step – VIII → While i<p_num1
                tag[process]=tag[process] -1;
                If tag[process]=0 then move this process at the head of the queue Repeat
Step – IX → goto step – III
Step – X → if tag[process] ==0 then step–XII else step – XI
Step – XI → Select min burst[process] form ready queue
Step – XII → Select min load_per[VM] (least loaded and if more than one then least hop Time and VM capacity
indicate by the num field also more than the request size) VM from the VM pool and p_cur[VM] = p_cur[VM] +1;
load_per[VM] = p_cur[VM]*100/num[VM]
Step – XII → remove the selected process form queue and dispatch it to selected VM p_num= p_num -1;
Step – XIV → if new process is arrived add to the tail of the queue and p_num2= p_num2 + 1;
Step – XV → While i<p_num2
                tag[process]=tag[process] -1;
              If tag[process]=0 then move this process at the head of the queue
          Repeat
Step – XVI → goto step II
Step – XVII → Stop
```

Time complexity of proposed algorithm

Let n processes arrives in unit time then n time tag [process] of ready queue process is updated until it will reach to zero. Let n1 job is arrived in the Imp_Ready and n2 job is arrived in the W_Ready. Let bound waiting tag is m which is less than n. Let number of existing process in the Imp_ready queue is p1 and number of existing process in the W_Ready is p2 and number of process leaves the Imp_Ready queue is r1 and number of process leaves W_Ready queue is r2. Let number of VM is s. Let total existing process is p and leaved process is r.

Time required to update tag[process] = p1 (Number of Existing process in the queue) + p2 (Number of Existing process in the queue) + n (Number of processes in the queue) -r1 (Number of processes leaves the Imp_Ready) – r2 (Number of processes leaves the Imp_Ready) = p + n - r

Time required to find minburst process in Imp_Ready = [p1 (Existing process in ready queue) + n1 (process arrived in the ready queue) – r1 (process leave the queue)] * r
Time required to find minburst process in W_Ready = [p2 (Existing process in ready queue) + n2 (process arrived in the ready queue) – r2 (process leave the queue)] * r

Time required to find the least loaded VM = s

Time requied to update the load_per[VM] = r (Number of process leaves the ready queue)

$$\begin{aligned}\text{Total time} &= p + n - r + (p1 + n1 - r1)*r + (p2+ n2-r2)*r + s + r\\ &= p + n - r + (p + n + r)*r + s + r\\ &= p + n + p*r + n*r - r^2 + s\\ &= p (r+1) + n(r + 1) - r^2 + s\\ &= (p + n) (r+1) - r^2 + s;\ n\&r >> s\end{aligned}$$

So Total time = $(p+n) (r+1) - r^2$
p is approximately equal to n
Then total time = $2n (r+1) - r^2$
Case – I n >> r
It means that number of job arrived is much more than number of job leaved

Then total time = O (n^2)
Case – II n is approximately equal to r
It means that number of job arrived is approximately equal to number of job leaved

$$\begin{aligned}\text{Then total time} &= 2r (r+1) - r^2\\ &= 2 r^2 + 2r - r^2\end{aligned}$$

Total time = O (r^2)

III. CONCLUSION AND FUTURE WORK

In this thesis we developed algorithm for private cloud to handle load balancing with effective scheduling algorithm and also ensure execution of important task as early as possible. For private cloud we develop algorithm which use SJF with bounded waiting and also consider the issue of load balancing and important job execution. Result of private cloud shows that less loaded VM is chosen for the execution of user request which will in turn increase the throughput of private cloud Private cloud work is good for the organization who wants to create their own setup to provide cloud computing to its user.

In future we create algorithm which a lot of priority because in this algorithm we have just two priority of job one is for important and zero is for less important and we use two different queue to manage the execution of these task but with increment in the size of organization and with increasing variety of users of organization we have to develop system which can handle lot of priority level because a lot of priority level is not handled by increasing number of queues or we can say that increase the number of queue to handle the number of priority level is not the feasible way .

REFERENCES

[1] Rashmi K S and Suma V and Vaidehi M "Factors Influencing Job Rejection in Cloud Environment", IJC May 2012.

[2] Rashmi K S and Suma V and Vaidehi M “Enhanced Load Balancing Approach to avoid Deadlocks in Cloud “IJCA-ACCTHPCA, June 2012

[3] Saurabh K Garg,Chee Shin Yeo, Arun Anandasivam, Rajkumar Buyya “ Environment – Conscious Scheduling of HPC application on distributed Cloud – oriented centers “ ScienceDirect My 2010

[4] Gabriel Mateescu , Wolfgan Gentzsch , Calvin J Ribbens “ Hybrid Computing – Where HPC meets grid and Cloud Computing “ ScienceDirect Nov 2010

[5] Weiwei Lin , james Z. Wang, Chen Liang, Deyu Qi “A Thrseshold – Based Dynamic Resource Allocation Scheme for Cloud Computing”, ScienceDirect 2011

[6] Raul Alonso-Calvo, Jose Crespo, Miguel Garcia- Remesal, Alberto Aungita and Victor Maojo “On Distributing load in cloud computing: A real application for very –large image datasets”, ScienceDirect 2010

[7] Ashraf Zia & Muhammad Naeem Ahmad Khan “Idetifying Key Challenges in Performance Issues in Cloud Computing”, IJMECS 2012

[8] Alexey Tumanov, James Cipar and Michael A Kozuch “Algebraic Scheduling of Mixed Workloads in Hetrogeneous Clouds, ACM 2012

[9] Monir Abdullah , Mohamed Othman “ Cost – Based Multi – QoS Job Scheduling using Divisible Load Theory in Cloud Computing “ ScienceDirect 2013

[10] R. Santosh and T. Ravichandran “Non-Preemptive on-Line Scheduling of Real-Time Services with Task Migration for Cloud Computing”, EJSR 2012

[11] Soumya Ray and Ajanta De Sarkar, “Execution Analysis of Load Balancing Algorithm in Cloud Computing Environment in “International Journal on Cloud Computing: Service and Architecture (IJCCSA), Vol 2 Oct 2012

[12] K. L. Giridas , A Shajin nargunam, “CHPS in Cloud Computing Environment “in International Journal of Engineering and Technology (IJET) Oct – Nov 2012.

[11] Zhang. Y and Zhou Y “TransOS: A Transparent computing-based operating system for the cloud”.

[12] Donald Mclaughin and Partha Dasgupta “Preemptive Scheduling for Distributed System”,

[13] Harsora and Dr. Apurva Shah, “A Modified Genetic Algorithm for Process Scheduling in Distributed System “IJCA, 2011

[14] Indraveer Chana and Anju Bala “A Survey of Varoous Workflow Scheduling Algorithm in Cloud Environment “NCICT 2011

[15] Anthony T. Velte, Toby J. Velte, Robert Elsenpeter, Cloud Computing a Practical Approach, TATA McGRAW-HILLEdition2010.

Authors Profile

Ashish Kumar Singh completed the B.E. in Computer Science from RGPV, Bhopal and M.B.A. from BU, Bhopal degrees in 2003 and 2007, respectively. Currently he is doing dissertation of M.E. (CS) final semester on same topic as of paper, From SRIT, Jabalpur, MP, India.
E-Mail: amsaidaitittc@gmail.com

Sandeep Sahu completed the M.Tech. in Computer Science from IIT Guwahati, India in 2009. Currently he is working with SRIT, Jabalpur as HOD, Department of Computer Science and Applications. He is the guide of dissertation work of Ashish Singh
E-Mail: sandeep.sahu12@gmail.com

Mangal Nath Tiwari completed the M. Sc. (IT) and M. Phil (CS) Degrees from APS University, Rewa in 2004 and 2010, respectively. Currently he is a student of Ph.D. in APS University, Rewa, MP, India.
E-Mail: mangal.tiwari81@gmail.com

Kamalendra Kumar Gautam completed MSc(IT) from MCU,Bhopal and M.Phil(CS) from APSU, Rewa, MP. Currently he is a student of Ph.D. in APS University, Rewa, MP, India.
E-Mail: kamalendrakumargautam@gmail.com