A SCHEDULING TECHNIQUE FOR QOS SENSITIVE JOBS IN CLOUD

S.P.Jeno Lovesum [#1], K.Krishnamoorthy [*2], Blessed Prince. P [#3]
[#] Department of Computer Science and Engineering-Department of Information Technology, Karunya University-Karunya University,
Karunya University,Karunya Nagar, Coimbatore,Tamilnadu, India
[1]jenolovesum@gmail.com
[3]blessedprince@gmail.com
[2] Department of Computer Science and Engineering, Sudharsan College of Engineering
Sudharsan College of Engineering, Sathiyamangalam, Pudukottai, Tamilnadu, India
[2]kkr_510@rediffmail.com

Abstract— The aim of this paper is to introduce a new algorithm for QOS based resource scheduling. Cloud is a type of parallel and distributed system consisting of a collection of inter-connected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements established through negotiation between the service provider and consumers. Resource scheduling, which is a part of resource management is an important process that takes place in storage cloud which falls under IaaS cloud, so that the available resources may be properly allocated to the requesting tasks in a best fit manner so that no resources are wasted. In this paper a new algorithm is designed for task scheduling to minimize memory wastage since our task is to store some files in the storage cloud, task completion time and task response time. The task manager checks all the virtual machine and assigns the task to proper virtual machine which will have least memory wastage.

Keywords—Cloud, Optimal, Scheduling, Storage Cloud, IaaS

I. INTRODUCTION

Cloud is a type of parallel and distributed systems consisting of a collection of interconnected and virtualized computers. They are dynamically provisioned and represented as one or more computing resources based on service level agreements, established between the service providers and consumers which involves negotiation. Cloud computing delivering hosted services over the Internet.

By using the virtualization concept, cloud computing can also support heterogeneous resources and flexibility is achieved. Another important advantage of cloud computing is its scalability. Cloud computing has been under growing spotlight as a possible solution for providing a flexible on demand computing infrastructure for a number of applications. All these factors increased the popularity of cloud computing. The services are broadly divided into three categories - Infrastructure-as a Service (IaaS), Platform as a Service (PaaS) and Software-as a Service (SaaS). Cloud computing is an entirely internet-based approach where all the applications and files are hosted on a cloud which consists of thousands of computers interlinked together in a complex manner. The processing units in cloud environments are called as virtual machines. It should make sure that the tasks are not loaded heavily on one VM and some VMs do not remains idle and/or under loaded. Load balancing of non-pre-emptive independent tasks on virtual machines is an important aspect of task scheduling in clouds. When a VM is overloaded and remaining VMs are under loaded with tasks for processing, the load on the VMs has to be distributed uniformly to achieve better CPU utilization. There are various algorithms designed for balancing the load among different tasks.

The goal of the load balancing is to optimize resource usage, increase throughput, reduce response time, and avoid overloading one of the VM. The processing units in cloud environments are called as virtual machines. Make sure that virtual machines should not be overloaded, less loaded or idle without tasks [1]. Scheduler should perform efficiently in this case. So load balancing of tasks is an important function of scheduler. Load balancing is used to achieve optimal machine utilization.

Load balancing techniques are of two types based on the current state of the system

1. Static
2. Dynamic

Static algorithms are used when there is low variation in the load. Static algorithms are not suitable for cloud environments because in cloud environment load is varying at varying time. Dynamic load balancing algorithms overcomes this drawback. Dynamic techniques are successful for load balancing of tasks among heterogeneous resources [2].

Load balancing is an important factor to improve the performance of the system. Based on process origination load balancing algorithm classified into

a) **Sender Initiated**: In this algorithm client sends request until receiver assigned to him to receive load.
b) **Receiver Initiated**: In this receiver send request to acknowledge the sender.
c) **Symmetric**: It is a combination of both receiver and sender initiated.

II. RELATED WORKS

There are various techniques that has been used for load balancing in cloud computing. Some of the techniques that has been used earlier are discussed in this chapter.

Vlad Nae et al. [3] bestowed event-driven load balancing algorithm for real time called as Massively Multiplayer on-line Games (MMOG). the MMOG machinist over-provision an own multi-server infrastructure with adequate competence for assuring the Quality of Service (QoS) requirements and a smooth game play at all times. The new cloud computing technology based on resource virtualization has the potential to provide an on-demand infrastructure for MMOGs, where resources are provisioned and paid for only when they are actually needed. The advantage of this techniques are i) it is capable of scaling a game session on multiple resources in keeping with the variable user load and ii) it conjointly provides rise to occasional QoS breaches.

In [4] the authors propose to find the best EFFICIENT cloud resource by Co-operative Power aware Scheduled Load Balancing solution to the Cloud load balancing problem. The algorithm uses the inherent efficiency of centralized approach, energy efficiency and the fault-tolerant nature of the distributed environment like Cloud H. Liu et al. [5] projected a load balancing on virtual memory strategy (LBVS) that gives an outsized scale web information storage model and Storage as a Service model supported Cloud Storage. This work basically defines a load balancing in virtual storage strategy and use Fair-Share Replication (FSR) and a executing a balancing algorithm to archive it. Since bandwidth and CPU speed are usually expensive to change, the another alternative way is by placing replicas of data objects closer to clients is the cheapest way. The main idea of FSR is to identify best candidate nodes for replica placement primarily based on access load. Storage virtualization is achieved using a schema of three-layered and load balancing is achieved using 2 load balancing modules.

A. M. Nakai et al. [6] projected a novel server-based load balancing policy for internet servers .The paper focuses on: (i) the evaluation of client-based server selection schemes in scenarios where several clients use the same schemes; and (ii) the proposal of a new solution that outperforms existing ones by dynamically adapting the fraction of load each client submits to each server.

SrinivasSethi et. al. [7] has introduced the novel load equalization technique using fuzzy logic in cloud computing, within which load equalization could be a core and difficult issue in Cloud Computing. In this work, the authors have designed a new load balancing algorithm based on round robin in Virtual Machine (VM) environment of cloud computing in order to achieve better response time and processing time. The load balancing algorithm is done before it reaches the processing servers the job is scheduled based on various parameters like processor speed and assigned load of Virtual Machine (VM) and etc. It maintains the information in each VM and numbers of request currently allocated to VM of the system. It identify the least loaded VM, to allocate the task and it identifies the first one if there are more than one least loaded machine.

Tayal [8] has proposed an optimized protocol based on Fuzzy-GA improvement that makes a programming call by evaluating the whole cluster of task within the job queue. In this work, the authors have described and assessed Fuzzy sets to model inexact scheduling parameters and also to symbolize satisfaction grades. The algorithms with different components are designed for task level scheduling in Hadoop Map Reduce. In order to achieve a better balanced load across all the nodes in the cloud environment, the scheduler is enhanced by forecasting the execution time of tasks assigned to certain processors and making an optimal decision over the entire group of tasks. This work also assumes centralized scheduling policy where a master processor collects all tasks, will also dispatch them to other process units. The system architecture illustrates the data store and computing cluster that jobs could be allocated to the cluster includes machines arranged in a general tree-shaped switched network.

Shamali Ambike et *al.* [9] propose a system for scheduling the multiple requests. Multiple requests are proposed by the use of non-pre-emptive priority algorithm. The algorithm implements the static load balancing technique based on the size of the files. This strategy aims to provide optimistic value of service and achieve an affirmative response at the users end.

Jiayin Li et *al.* have discussed about an adaptive resource allocation for preemptable Jobs in Cloud Systems with two task scheduling algorithms like Adaptive list scheduling (ALS), Adaptive min-min scheduling (AMMS) [10] [11]. This algorithm adjusts the resource allocation adaptively based on the updated actual task execution times but their approach does not pertain to cost optimization and time optimization. Load balancing ensures that all the processor in the system or every node in the network does approximately the equal amount of work at any instant of time. A comparison is also made between load balancing strategies.

In [12], Meenakshi Sharma et *al.* have analyzed efficient load balancing algorithm in VM cloud environment in order to achieve better response time and cost.

The authors of [13] presented QoS based priority scheduling algorithm for task scheduling in cloud environment. In this algorithm, cloud scheduler first allocates the important (higher priority) tasks to the resources and then it allocates the low prioritize task so as to achieve the maximum resource utilization rate, minimize the make span and avoid the load balancing level problem. QoS Priority Based Scheduling Algorithm gives results better than QoS guided weighted mean time min,Min-Min and Max- Min heuristic algorithms**.**

Nathani et *al.* [14] recommended deadline sensitive resource scheduling in IaaS cloud computing by experimenting in Haizea scheduler while minimizing the total number of leases rejected by it. Dynamic planning based scheduling algorithm has been implemented in Haizea scheduler by the authors, which can admit new leases and prepare the schedule whenever a new lease can be accommodated.

Akhani et *al.* [15] proposed decision making algorithms and extended the current Advanced Reservation (AR) algorithms in Haizea scheduler to provide negotiation based allocation of resources.

The literature survey reveals the different ways of task scheduling and load balancing approaches. This paper proposes the new efficient task scheduling algorithm for QOS sensitive jobs. As in our work we have focused on storage cloud which involves memory space we try to reduce the wastage of memory space in the VM by scheduling the tasks among the available resources. The algorithm adopts a new method of classifying the tasks into high priority, low priority and medium priority jobs based on some threshold values.

III. ARCHITECTURE AND ALGORITHM

A. *Architecture:*

As in our work we have focused on storage cloud which involves memory space we try to reduce the wastage of memory space in the VM by finding the best resource using the scheduling algorithm and allocating the task to that resource. Most algorithms used for scheduling and load balancing considers make span, response time, throughput, deadline, priority as their performance metric or the QOS factors. But in algorithm we have considered the wastage of memory space as the QOS parameter for scheduling. This scheduling method will assign to the virtual machine in best fit manner. i.e., task manager(TM) will check all the virtual machine and assigns the task to proper virtual machine which will have least memory wastage.

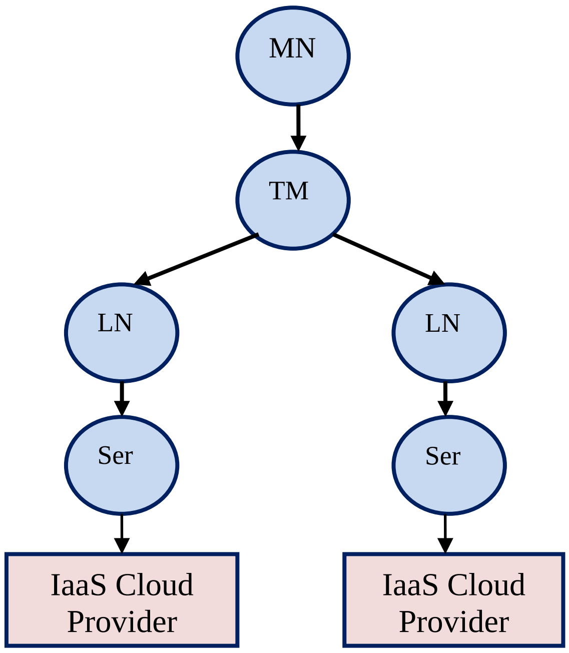

Fig.3.1. Overall architecture of the proposed method

User sends their task request (for e.g., storing a file in the storage cloud) to the cloud server. The task manager in the cloud server will decide which virtual machine to allocate to store that task. Components of TREE are Master Node (MN), Task Manager (TM), Local node(LN) and Server (Ser). It has a hierarchical structure with one MN, which is the root of the hierarchy. Users are directly contact with MN. TREE architecture allows direct connection with IaaS cloud. IaaS cloud will be connected to the Server nodes, who will decide when to allocate and release resources according to the users' demand. Each service will be run in the VMs hosted in the cloud. Once a VM is created, the Ser node will be connecting to the corresponding VM in order to execute user tasks. While receiving a task request from the user, the request travels through the TREE hierarchical architecture and reaches at Ser. Before computing the offers, TM will check some preliminary conditions. That means, it will check whether the received resource request is available with the cloud providers. In our system, it will first check active VMs, then inactive VMs and at last in new VMs. Here the resource considered for the experiment is CPU and memory. The request will contain the amount of CPU needed (expressed in MIPS),and memory space in bytes.

As mentioned earlier, before computing the offers, preconditions have to be satisfied. That means, first TM has to check whether the requested amount of CPU and memory space is available in the active VM, then it will check in the inactive VM. If these two options are not satisfying, it will go for starting a new VM. In this case, two conditions have to be satisfied. First, it has to check that the VM has the capacity to accomplish the request. Next, it has to check that the new VM can be deployed in a physical server or host.

If the conditions are satisfying, the offers have to be calculated. That means, the TM has to calculate the time required to complete a particular requested task. Let's denote the requested resource using $r_{i.}$ The processing time taken for the particular task request is calculated using the given formula:

$$P_t = r_i/C_j \tag{1}$$

Where, C_j represents the processing speed of the particular VM. In the case of active VMs, some processes will be running already in that VM. Consider both the currently running task and the task residing in the VM's queue. The processing time in this case is calculated as follows:

$$P_t = (r_i+R_i)/C_j \tag{2}$$

Where, R_i represents the sum of all task execution time and the remaining execution time of the current task. In the last case, we have to find the processing time in a new VM.

This is calculated by adding a fixed VM starting time with the equation 1. After completing the processing offers calculation, the offers are sent through the TREE hierarchy upto the MN. When the offer reaches MN, it will be forwarded to the corresponding users. The user will select appropriate offers from the offer list based on their preference. After selecting the appropriate offer, user will contact the corresponding VM.

B. *Algorithm*

In order to handle QOS sensitive jobs, new algorithm is developed. This is called QOS Based Scheduling (QBS). Algorithm steps are given below:

```
Step 1: User sending requests with four parameters: RequestID ($R_{id}$), Timestamp (TS), Deadline (DL), Memory space(MS)
Step 2: Requests reaches TAM(Task allocation manager in the SP(service provider) through TREE architecture.
Step 3: Sort the available VMs based on their processing capacity and storage capacity
Step 3: TAM computes the propagation time ($P_{time}$)
Step 4: Compute new deadline for task execution $AV_{time}=DL-p_{time}$
Step 5: If $AV_{time} > TH_u$
            Insert the task into the low priority array and goto step 6.
        Else if $TH_u > AV_{time} > TH_l$
            Insert the task into the high priority array and goto step 7
        Else if $AV_{time} < TH_l$
            Execute the request immediately
Step 6: Sort the lower priority queue and compute the allocation offers on the low capacity VMs. By doing this energy efficiency is acquired.
Step 7: Sort the higher priority queue and compute the allocation offers on the high capacity VMs. By doing this energy efficiency is acquired.
```

In this algorithm, the request reaches the TM it will process the requests. It will maintain three separate arrays for execution. One is the lower priority array, another one is the higher priority array and the last one is urgent array. When the request reaches the TM, first it will calculate the propagation delay. Propagation delay is calculated by subtracting the current time with the timestamp attached with the request. After the calculation of propagation delay, the propagation delay is subtracted from the deadline associated with the task request. The result is the available time for the task execution before deadline. Priority is set based the QOS parameters specified by the user.

Next, the tasks have to be sorted based on their available deadline. Tasks are sorted based on some threshold value. If the tasks are below the lower threshold value, the tasks are moved to urgent array. It will hold all the urgent tasks. Next the array is sorted based on their deadline value. These tasks are executed immediately without any delay.TM won't compute the allocation offers for these tasks and sent back to the user. For these types of tasks, it will just send an acknowledgement to the user and execute the task.

First the urgent tasks are sent for the execution in VMs. After that medium important tasks are serviced. To find such tasks, tasks are filtered whose value lies between the upper threshold and lower threshold. These tasks are stored in the higher priority array. Tasks stored in this array will compute the allocation offers. Offer calculation for these tasks is given higher priority. These task offers are calculated on the high capacity VMs. This will helps the user to execute the tasks without missing the deadline.

Finally, normal tasks have to be served. In this case, tasks are categorized based on their threshold value which is less than the lower threshold value. As these tasks are not so much concerned about the deadline, TM will calculate the offers in low capacity VMs. For the tasks other than the urgent tasks, offers are calculated by the TM and the offers are sent back to the users. When the offers reach the user, it will select the most appropriate offer and contact the VM directly.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

The system is implemented using CloudSim. This experiment is conducted with 20 tasks. Tasks are randomly assigned to the resources. Every task is assigned with a deadline, time needed for the completion of the task and memory space needed. Tasks are failed mainly due to few reasons. One, there is no enough resources available to complete the task. Second one is the processing offers send a processing time that is greater than the deadline of the corresponding tasks.

The experiment is performed and compared with two different algorithms. First, the task requests from users are served based on the First Come First Server (FCFS) basis. Next is based on the Earliest Deadline First (EDF) Basis. The above two are commonly used techniques for scheduling. In order to handle QOS sensitive jobs, a new algorithm is implemented in this paper.

According to QBS, the total tasks and failed tasks are analysed. As per this algorithm, the experiment is performed. Out of the 20 tasks, 18 are completed without any problems. 2 tasks are marked as failed. In the final output it showed as that some tasks are failed due to different reasons. Overall percentage of the completed tasks is 90. Graphical representation of all the tasks is represented in Fig. 4.1. It shows the overall representation of the tasks executed in the system.

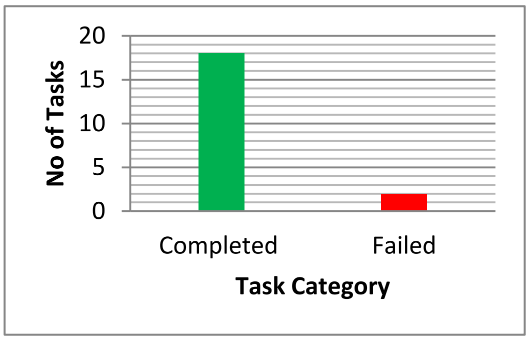

Fig. 4.1 Total task analysis

Next the failed tasks are analysed in detail. Two tasks are failed in this experiment. Task failure reasons are divided into three. First one is no offer received. That means within the deadline time, no suitable offers are received at the user side. Second reason is the task failure due to deadline missed. In this case, deadline can be missed in many cases like while sending the request through the hierarchy, while computing the offers, sending the offers back to the user through the hierarchy etc. Failed tasks analysis is shown in Fig. 4.2

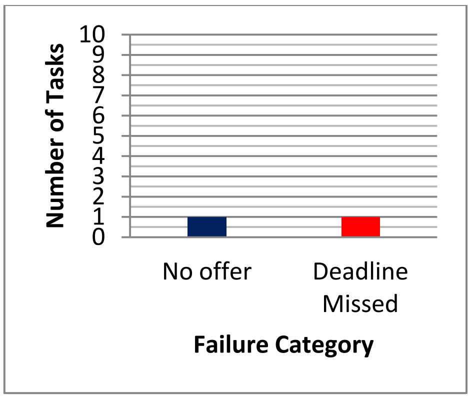

Fig. 4.2 Failed task analysis

Performance is analysed based on the metric deadline miss rate which is the rate of the tasks missed their deadline. Deadline miss rate (DMR)is calculated as follows:

DMR =Tasks missed deadline/Total task requests (3)

Deadline miss rate is calculated for the three algorithms and the results are shown in table 1.

Table 1 Comparison of Different Algorithms

ALGORITHM	DEADLINE MISS RATE
FCFS	.25
EDF	.15
QBS	.05

V. CONCLUSION AND FUTURE WORK

In this paper a QOS based scheduling technique has been discussed and implemented. The QoS factor considered here is the memory space wastage in storage cloud which protects the cost of the cloud user and the cloud service provider and the deadline specified by the user for a particular job. Performance is evaluated based on the deadline miss rate parameter. For every task, the QOS parameters are specified. Scheduling is done based on the deadline specified by the user for the request reply and the QOS specified by the user. As future work this scheduling technique may be optimized using PSO so that an optimum resource scheduling can be done which will reduce the cost of the user in using a particular resource and the cost of the service provider in allowing a user to use its resource.

REFERENCES

[1] DhineshBabu L.D, P. VenkataKrishna ,"Honey bee behavior inspired load balancing of tasks in cloudcomputing environments",2013
[2] NayandeepSran, NavdeepKaur,"Comparative Analysis of Existing Load Balancing Techniques in Cloud Computing ",2013.
[3] Vlad Nae, Alexandru Iosup, Radu rodan," Dynamic ResourceProvisioning in Massively Multiplayer Online Games", Transactions on Parallel and Distributed Systems,2010.
[4] T V R Anandarajan, M A Bhagyabini, "Co-operative scheduled Energy aware load-balancing technique for an efficient computational cloud", IJCSI, volume 8, issue 2,2011
[5] Hao Liu, Shijun Liu, Xiangxu Meng, Chengwei Yang, Yong Zhang,LBVS"A Load Balancing Strategy for Virtual Storage, IEEEInternational Conference on Service Sciences,2010.
[6] Alan Massaru Nakai, Edmundo Madeira, and Luiz E. Buzato," Improving the QoS of Web Services via Client-Based Load Distribution",2011
[7] Srinivas Sethi, Anupama Sahu, Suvendu Kumar Jena, "Efficient load Balancing in Cloud Computing using Fuzzy Logic", IOSR Journal of Engineering (IOSRJEN) ISSN: 2250-3021 Volume 2, Issue 7,PP 65-71,2012
[8] Sandeep Tayal, "Task Scheduling Optimization for the Cloud Computing"2011
[9] Shamali Ambike, Dipti Bhansali,Jaee Kshirsagar,Juhi Bansiwal,"An Optimistic Differentiated Job Scheduling System for Cloud Computing", International Journal of Engineering Research and Applications (IJERA), Vol. 2, Issue 2,2012
[10] Jiayin Li, Meikang Qiu,Jian-Wei Niu, Yu Chen, Zhong Ming,." Adaptive Resource Allocation for Preemptable Jobs in Cloud Systems", IEEE International Conference,2010
[11] Jiayin Li, Meikang Qiu, Zhong Ming, Gang Quan, Xiao Qin, Zonghua Gu." Online optimization for scheduling preemptable tasks on IaaS cloud systems" Journal of Parallel and Distributed Computing,72,666-677,Elsevier ublications,2012
[12] Meenakshi Sharma,Pankaj Sharma,Dr.Sandeep Sharma," Efficient Load Balancing Algorithm in VM Cloud Environment", International Journal of Computer Science and Technology(IJCST), Vol. 3, Issue 1,2012.
[13] T Amudha, T T Dhivyaprabha," QoS Priority Based Scheduling Algorithm and ProposedFramework for Task Scheduling in a Cloud Environment" IEEE International Conference on Recent Trends in Information Technology (ICRTIT),2011.
[14] A.Nathani, S.Chaudhary, and G.Somani, "Policy based resource allocation in IaaS cloud",Future Generation Computer Systems,2011.
[15] J.Akhani, S.Chaudhary, and G.Somani, "Negotiation of Resource Allocation in IaaS Cloud,Proc. Fourth Annual ACM Bangalore Conference, Bangalore, India,2011.