k-TIER AND SELECTIVE BACKFILLING APPROACH FOR PARALLEL WORKLOAD SCHEDULING IN CLOUD

Shahabanath K K, Sreekesh Namboodiri T

Abstract— **Cloud Computing is known as a provider of very large scalable dynamic services and virtualized resources over the Internet. Job scheduling is most important task in cloud computing environment because user have to pay for used resources based upon time. The main goal of scheduling is distribute the load among processors and maximizing their utilization by minimizing the total task execution time and also maintaining the level of responsiveness of parallel jobs. Existing parallel scheduling mechanisms have some drawbacks such as context switching rate, large waiting times and large response time. There exists two-tier priority based consolidation methods : conservative migration and consolidation supported backfilling and aggressive migration and consolidation supported backfilling. In this method, partition the computing capacity of each node into two tiers, the foreground virtual machine (VM) tier (with high CPU priority) and the background VM tier (with low CPU priority). In this paper propose a new scheduling method which is better than both conservative and aggressive backfilling. This method provide reservation selectively, only to the jobs that have waited long enough in the queue. Also divide the computing capacity into k-tier. There will be one foreground VM and number of background VMs.**

***Index Terms*—- Aggressive Backfilling, Conservative Backfilling, Reservation, Workload Consolidation.**

I. INTRODUCTION

Cloud computing is an on demand service in which shared resources, information, software and other devices are provided according to the clients requirement at specific time. Scheduling jobs is an important issue in the cloud. There are various scheduling algorithm exist in cloud computing environment.The main goal of scheduling algorithm is distribute the load among processors and maximizing their utilization while minimizing the total task execution time. The job scheduler is responsible for assigning preferred resources to a particular job so that the overall computing resources are utilized effectively. The application also has to make sure each job is given adequate amount of resources, or its fair share.

Manuscript received September, 2014.
***Shahabanath K K**, Computer Science and Engineering, MES College of Engineering, Kuttippuram, India,*
***Sreekesh Namboodiri T**, Computer Science and Engineering , MES College of Engineering, Kuttippuram, India..*

There two types of scheduling :

- Application scheduling
- Job scheduling

The process of scheduling parallel tasks determines the order of task execution and the processor to which each task is assigned. Typically, an optimal schedule is achieved by minimizing the completion time of the last task. Two types of scheduling strategies are space sharing and time sharing.Time sharing tech- niques virtualizes the physical machine by slicing the time axis into multiple virtual machines. Space sharing techniques runs the jobs side by side on different nodes of the machine at the same time.

These cloud computing environments provide an infinite computing resources to cloud users so that they can increase or decrease their resource consumption rate according to the demands. Cloud providers hold massive computing resources in their large datacenters. The user have many applications and they lease resources from providers to run their applications. A user sends a request for resources to a provider. When the provider receives the request, it looks for resources to satisfy the request and assigns the resources to the requesting user, typically as in the form of virtual machines(VMs). Then the user uses the assigned resources to run applications and pays for the resources that are used. When the user is done with the resources, they are returned to the provider. Job scheduling is one of the major activities performed in all the computing environments. Cloud service scheduling is categorized at user level and system level. At user level scheduling deals with problems raised by service provision between providers and customers. The system level scheduling handles resource management within datacenter.

II. LITERATURE SURVEY

A. First-Come-First-Serve

The basic batch scheduling algorithm is First-Come-First-Serve (FCFS)[2]. Under this algorithm, jobs are considered in order of arrival. If there are enough processors are available to run a job, the processors are allocated and the job is started. Otherwise, the first job in the queue must wait for some currently running job to terminate. This may lead to a waste of processing power as processors sit idle waiting for enough of them to accumulate.

B. Backfilling

D. Feitelson al. [8] proposed a backfilling scheme. Backfilling is an space sharing optimization that tries to balance between the goals of utilization and maintaining FCFS order. It allows small jobs to move ahead and run on processors that would otherwise remain idle. This is done to avoid situations in which the FCFS order is completely violated and
some jobs are never run. There are two types of backfilling algorithms:

- Conservative Backfilling
- Aggressive Backfilling

1. Conservative Backfilling: Ahuva et al. [2] proposed a conservative backfilling approach. In this scheme, jobs are scheduled according to the order of arrival time when there is enough number of processors.If not, another job with later arrival time and smaller jobs are scheduled to run. It provides reservation to all jobs and limits the slowdown.
2. EASY Backfilling: Ahuva et al. [2] also proposed an aggressive approach, provides a reservation to only the job at the head of the job queue and only allow job at the head of the queue can be pre-empt other jobs. It does not have a guaranteed response time of the user job at the time of job submission.

C. Gang Scheduling

Moreiraz et al. [3] proposed a main alternative to batch scheduling is gang scheduling, that schedules related threads or processes to run simultaneously on different processors. It is a time sharing optimization technique. This scheduling is used if two or more threads or processes are communicate with each other. The problems with gang scheduling is that the
requirement that all a jobs processes always run together causes too much fragmentation and context switching overhead.
Gang scheduling is based on a data structure called ousterhout matrix. In this matrix each row represent a time slice, and each column represent a processor. The threads or processes of a job packed into arow of the matrix.

D. Backfilling Gang Scheduling

Moreiraz et al. [3] proposed a Backfilling gang-scheduling (BGS) method. It is an optimization technique which combines gang scheduling and backfilling scheduling. This scheduling can be done by treating each of the virtual machines created by gang-scheduling as a target for backfilling. Which produce better results than individual approaches gang scheduling or backfilling.

E. Migration Gang Scheduling

Moreiraz et al. [3] proposed a Migration Gang Scheduling method. The process of migration embodies moving a job to any row in which there are enough free processors to execute that job. There are two options for migrate a job from a source row to a target row.

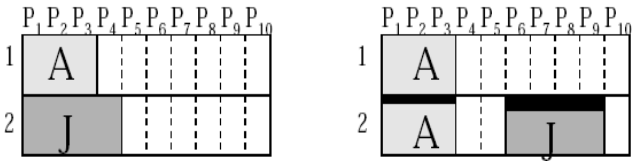

Fig. 1. Migration Option 1

In the Fig. 1, job A resides in the first row and job J in the second row occupy the same columns as job A in first row. Job J migrate to other columns in the same row and job A is replicated to second row.

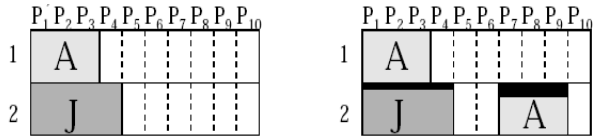

Fig. 2. Migration Option 2

In migration option 2, job A in the first row can be directly migrate to second row.

F. Job kill based EASY Backfilling (KEASY)

Xiaogang Qiu et al. [4] proposed a scheduling scheme Job kill based EASY backfilling (KEASY). It is capable of dispatching a job to run in background VMs while it is not qualified for backfilling according to EASY. There is a chance that the corresponding foreground VMs of those background VMs are idle during the jobs lifetime, which leads to performance improvement.

G. Reservation based EASY Backfilling (REASY)

Xiaogang Qiu et al. [4] proposed a Reservation based EASY backfilling (REASY) scheme. In this scheme job kill is not allowed in the scheduling; once a job is deployed onto background VMs of a set of pro- cessors, its run is pinned onto this set of processors. Only all the foreground VMs of this set of processors are available can this job run in the foreground. For the reservation making, if a reservation is being made for a job running in the background tier, the shadow time is the last termination time of the jobs running in its foreground VMs; the extra foreground VMs are the ones now idle and no process of the job is running in their background VMs.

H. Conservative Migration Supported Backfilling

Xiaocheng Liu et al. [5] proposed a Conservative Migration Supported Backfilling (CMBF), which is same as Conservative backfilling. Only the difference is, the scheduler is able to suspend a job and resume it on other nodes in a later time. This algorithm avoid starving a pre-empted job. When the number of jobs in the queue is large, the cost can be high because CMBF requires tracking backfilling jobs for each job in the queue.

I. Aggressive Migration Supported Backfilling

Xiaocheng Liu et al. [5] proposed an An alternative to CMBF is Aggressive Migration Supported Backfilling (AMBF). It only tracks backfilling jobs for the job at the head of the queue and allows the head-of queue job to pre-empt other jobs. The rest of jobs in the queue are not allowed to pre- empt jobs.

J. Conservative Migration and Consolidation supported BackFilling

In priority-based method to consolidate parallel workloads in the cloud, dividing computing capacity into two tiers : foreground tier and background tier[6]. The VM running in foreground is assigned a high CPU priority and the VM running in background is assigned a low CPU priority. There are two priority-based methods to consolidate parallel workloads in the cloud: Conservative migration and consolidation supported backfilling (CMCBF) and Conservative migration and consolidation supported backfilling (AMCBF).

Xiaogang Qiu et al. [5] proposed a priority based consolidation method, Conservative Migration and Consolidation supported BackFilling (CMCBF). That allows jobs to run in background VMs simultaneously with those foreground VMs to improve node utilization. It ensures that a job is dispatched to foreground VMs whenever the foreground VMs are idle or that job satis_es the node requirement. It allows jobs to run in background VMs simultaneously with those foreground VMs to improve node utilization. Compared to CMBF, CMCBF also deals with how to ensure that the background workload does not affect the foreground job. CMCBF only dispatches a job to run in background VMs when the corresponding foreground VMs have a utilization lower than a given threshold. The foreground VM utilization can be obtained from the profile of foreground jobs, or from the runtime monitoring data.

III. AGGRESSIVE MIGRATION AND CONSOLIDATION BASED BACKFILLING

Aggressive Migration and Consolidation supported BackFilling[5] is a parallel workload consolidation scheduling which take the advantage of virtualization technology. This scheduling method has a two tier architecture to consolidate parallel workload in a datacentre and give a scheduling algorithm to optimize the response time of parallel jobs. The two tier VMs have different CPU priorities in a physical processor. The one with the high CPU priority is called foreground VM and the one with the low CPU priority is called background VM. A foreground VM and a background VM are pinned to one physical processor. They can simultaneously process different jobs.

By using this algorithm, schedules jobs to run according to their arrival time when there is enough number of nodes. When the number of idle nodes is not sufficient for a job, another job with a later arrival time but smaller node number requirement may be scheduled to run via backfilling.

A pre-empted job which should be the head of the queue is scheduled to run whenever it sees the total number of nodes that are either idle or occupied by jobs with a later arrival time is equal or greater than the number of nodes it needs. The scheduler instructs these jobs to save states, suspends their execution, and moves them back to the job queue. AMCBF only tracks backfilling jobs for the job at the head of the queue and allow the head-of queue job to pre-empt other jobs. The rest of jobs in the queue are not allowed to pre-empt jobs, in another word, they can only be dispatched to idle nodes.

It ensures that a job is dispatched to run in foreground VMs whenever the number of foreground VMs that are either idle or occupied by jobs arriving later than it satisfies its node requirement. Meanwhile, it allows jobs to run in background VMs simultaneously with those foreground VMs to improve node utilization. only dispatches a job to run in background VMs when the corresponding foreground VMs have a utilization lower tha a given threshold.

When a job is departure from background tier, the next job dispatched to background tier if the number of nodes needed by that job is less than or equal to the idle nodes in the background tier. The nodes needed by a new arrival job is less than or equal to the foreground idle nodes, it dispatched to the foreground tier. Otherwise it is dispatched to the background tier if there is enough processors are available.

IV. PROBLEM DEFINITION

CMCBF method requires tracking backfilling jobs for each job in the queue when making pre-emption decisions. When the number of jobs in the queue is large, the cost can be high. Simplify this algorithm called AMCBF to address this problem. In AMCBF, only tracks backfilling jobs for the job at the head of the queue and allows the head-of-queue job to pre- empt other jobs. The rest of jobs in the queue are not allowed to pre-empt jobs. In AMCBF, there is a delay in the execution of jobs other than first job in the queue. This algorithm challenging to achieve responsiveness of parallel jobs and high processor utilization in the cloud. Another issue in a large data center is the communication cost is high because the processes of a job to be allocated to nodes may not be close to each other.

V. K-TIER AND SELECTIVE BACKFILLING APPROACH

Effective scheduling schemes are important in cloud computing environment to improve node utilization and user metrics like slowdown and turnaround time. The use of the backfilling in job scheduling results in signi_cant imrovement to system utilization over non-backfilling schemes. There are two variants of back_lling scheduling are : Conservative backfilling and Aggressive backfilling. With conservative backfilling, each job is given a reservation when it arrives in the queue and doesn't delay any job in the queue because a guaranteed start time is given to each job. In aggressive backfilling, it gives reservation only to the head of the queue. Here propose a k-tier architecture to consolidate

parallel workload in a datacenter and give a new scheduling algorithm to optimize the response time of parallel jobs which takes the advantage over both conservative backfilling and aggressive backfilling. The new method is selective backfilling which give reservation only for selected jobs.

A. k-tier Architecture

Cloud computing model attracts many complex applications that run on data centers. Complex applications often require parallel processing capabilities. By using virtualization technologies the computing capacity of each node is divided into K-Virtual Machines (KVMs). Attaining KVMs are based on the available resources of physical machine. Collocation of virtual machines on each node will run multiple jobs simultaneously. Running parallel jobs on each node would increase resource utilization on each physical machine.

i) Virtualization

Virtualization broadly describes the separation of a resource or request for a service from the underlying physical delivery of that service. Virtualization can be defined as the ability to run multiple operating systems on a single physical system and share the underlying hardware resources. Virtual server seeks to encapsulate the server software away from the server hard ware. The virtual server consists of the operating system, the storage and the application. If virtual servers are maintained, these servers are serviced by one or more hosts and on host can maintain more than one virtual server. Same as traditional servers these also named according to their usage. By maintaining the Virtual servers it is possible to reduce their services providing for them if the administrator feels that services providing by them crosses its limit then they will reduce it to certain level. And they will adjust them. In order to develop virtual servers, there are several templates so, that it is possible to built multiple and identical virtual servers. The main advantage of virtual servers is, they can be transitioned from one host to another host.

ii) k-tier Computation

Construction of cloud architecture consists of cloudlet, data centers, virtual machines, etc. Here each node will have k-VMs according to the capacity physical machine. There will be one foreground VM and number of background VMs. Foreground virtual machine will have high priority among the all other virtual machines on the node. Parallel application with dependency among its parallel processes, achieving high utilization on the nodes on which these processes run is often difficult. For a cloud service provider that runs this kind of applications, how to address this issue is important for its competitiveness in the market. The very first step is to select nodes which are all having free VMs on it. By selecting these nodes, we can check CPU utilization of foreground virtual machine and if it is lower than a threshold then the incoming job to background virtual machine of the respective node. It also provides an opportunity to change jobs which are running on the background to foreground when there is enough resource to run.

B. Selective Backfilling

The selective backfilling scheme takes best characteristics from both conservative backfilling and aggressive backfilling while avoiding drawbacks. In this method, provide reservation only to the job who is wait long enough in the queue. According to this scheduling strategy, jobs are not given a reservation(i.e. their start and finish times are not defined or they are not used for backfilling) until their expected slowdown exceeds some threshold, whereupon they get a reservation. In other words, if a job waits enough, then it is given a reservation.

In other words, if a job waits enough, then it is given a reservation. This is done when the eXpansion Factor (XFactor) of the job exceeds some starvation threshold.

The XFactor of a job is defined as :

$$XFactor = (Waittime + Estimatedruntime) /Estimatedruntime \quad (1)$$

The slowdown for each job,

$$slowdown = (Waittime + runtime)/runtime \quad (2)$$

The XFactor threshold is the average slowdown of the completed jobs,

$$XFactorThreshold = max(10; averageslowdown) \quad (3)$$

C. Scheduling Process in Cloud

Scheduling in cloud is responsible for selection of best suitable resources for task execution. Scheduling process in cloud can be generalized into three stages:

1. Resource discovering and filtering : Datacenter Broker discovers the resources present in the network system and collects status information related to them.
2. Resource selection : Target resource is selected based on certain parameters of task and resource. This is deciding stage.
3. Task submission : Task is submitted to resource selected.

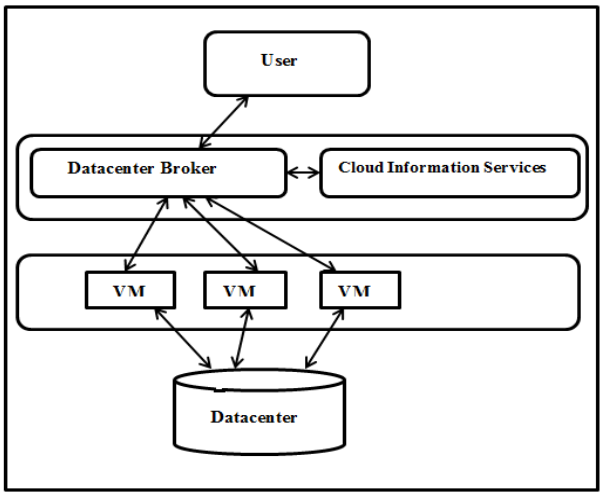

Fig. 3 . Scheduling Process

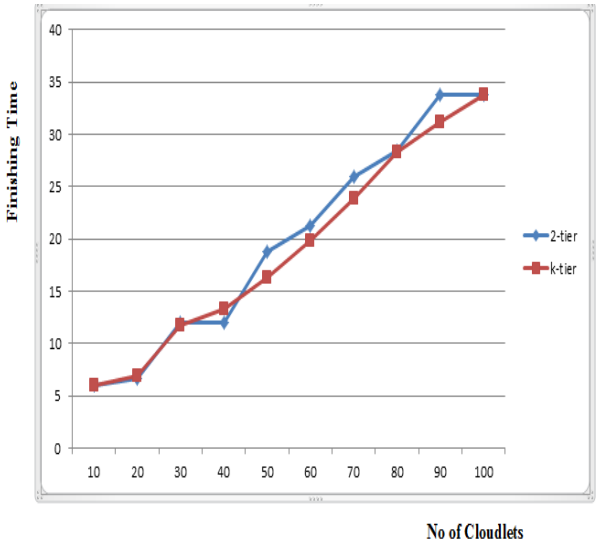

Fig. 4. Finishing Time of Cloudlets

Fig. 3 shows the scheduling process in the cloud. Cloud users or the Cloud consumers request for the VMs and ask for VM requirement to the Cloud broker. The Cloud broker or data center broker acts on behalf of the broker, looks for the Cloud service providers who can make VM request fulfilled by querying the Cloud Information Service (CIS). Every new resource should be register to the CIS. Once the Cloud service provider has been chosen by Broker, it submits the VMs list to the Datacenter. Datacenter holds the physical computing servers i.e hosts and VMs place on host on the basis of allocation policies decided by the Cloud service provider.

VI. Result and Discussions

CloudSim is used as a simulation tool with NetBeans 7.1 in this work. CloudSim is an extensible simulation toolkit that enables modeling and simulation of Cloud computing systems and application provisioning environments. The CloudSim toolkit supports both system and behavior modeling of Cloud system components such as data centers, virtual machines (VMs) and resource provisioning policies. In our simulation tool CloudSim tasks are modeled as Cloudlets and machines are modeled as VMs and cloud itself is modeled as a Datacenter. VMs are the virtual machines on which cloudlets are executed. These VMs are mapped onto hosts (Physical Machines) based on hardware requirements (processing cores, memory and storage). Processing capabilities of host are measured using MIPS (Million Instruction per Second).

Fig. 4 shows finishing time of the cloudlets. K-tier architecture shows slightly better perfomance than 2-tier architecture in terms of finishing time. In k-tier architecture, the node utilization is better than 2-tier architecture.
In this case, consider only one processing elements for each cloudlets and virtual machine. If there are more than one processing elements for each virtual machine, then the resource utiliation in k-tier archuitrecture is much better than in 2-tier architecture.

VII. Conclusion and Futureworks

The parallel applications sometimes shows the decreasing utilization of processors because of the communication and synchronization among parallel processes. There is a priority based Workload consolidation supported by virtualization is commonly used for improving utilization in data centres. The parallel workload consolidation method improves the responsiveness of of the job and minimize the execution time of the job. There are two different consolidation based algorithms for scheduling the workloads in the cloud: Conservative Migration and Consolidation supported Backfilling and Aggressive Migration and Consolidation supported Backfilling. There is a selective reservation strategy that is superior to both conservative backfilling and Aggressive backfilling. This new method takes the advantage of both methods. In selective reservation method, provide reservations only to the jobs which is waited long time in the queue. This reduce the waiting time of the jobs in the queue and also minimize the total task completion time. And improve the node utilization by divide the computing capacity of a node into k-tier. In future, exploit mechanisms that can effectively schedule the process among the intra-cluster of datacenter resources which may further improve the node utilization and responsiveness for parallel workload in the cloud.

REFERENCES

[1] David Jackson, Quinn Snell, and Mark Clement, "Core Algorithms of the Maui Scheduler", Workshop Job Scheduling Strategies for Parallel Processing, 2001.
[2] Ahuva W. Mu'alem and Dror G., "Utilization, Predictability, Workloads, and User Runtime Estimates in Scheduling the IBM SP2 with Backfilling", IEEE Transactions on Parallel and Distributed Systems , June 2001.
[3] Y. Zhangy H. Frankez J. E. Moreiraz A. Sivasubramaniamy, "An Integrated Approach to Parallel Scheduling Using Gang-Scheduling, Backfilling and Migration", IEEE Transactions on Parallel and Distributed Systems, March 2003
[4] Xiaocheng Liu, Bin Chen, Xiaogang Qiu, Ying Cai, and Kedi Huang, "Scheduling Parallel Jobs Using Migration and Consolidation in the Cloud", Mathematical Problems in Engineering, 2012.
[5] Xiaocheng Liu, Chen Wang, Bing Bing Zhou, Junliang Chen, Ting Yang, and Albert Y. Zomaya , "Priority-Based Consolidation of Parallel Workloads in the Cloud", IEEE Transactions on Parallel and Distributed Systems, September 2013.
[6] Xiaocheng Liu, Chen Wang, Xiaogang Qiu, Bing Bing Zhou, Bin Chen and Albert Y. Zomaya, "Backfilling under Two-tier Virtual Machines", IEEE Transactions on Parallel and Distributed Systems , June 2012.
[7] Ioannis A. Moschakis, Helen D. Karatza, "Performance and Cost evaluation of Gang Scheduling in a Cloud Computing System with Job Migrations and Starvation Handling", IEEE, 2011.
[8] D. Tsafrir, Y. Etsion, and D. Feitelson, "Backfilling Using System Generated Predictions Rather than User Runtime Estimates", IEEE Transactions on Parallel and Distributed Systems, 2007.
[9] Yogita Chawla and Mansi Bhonsle, "A Study on Scheduling Methods in Cloud Computing", International Journal of Emerging Trends Technology in Computer Science (IJETTCS), Volume 1, Issue 3, September October 2012.
[10] Pinal Salot, "A Survey Of Various Scheduling Algorithm in Cloud Computing Environment", International Journal of Research in Engineering and Technology(IJRET), Volume: 2 Issue: 2, ISSN: 2319 - 1163, February 2013.
[11] Gunho Lee, "Resource Allocation and Scheduling in Heterogeneous Cloud Environments", Electrical Engineering and Computer Sciences University of California at Berkeley, May 10, 2012.

Shahabanath K K received the bachelor's degree in Information Technology from the Calicut University, Kerala in 2012. Presently she is pursuing her M.Tech in the department of Computer Science and Engineering from Calicut University, Kerala. Her research interests include Load balancing in cloud, Scheduling in cloud etc.
Sreekesh Namboodiri T received bachelor's degree in Computer Science and Engineering from calicut university and master's degree in Computer Aided Design from TamilNadu. Currently working as an Assistant Professor in Computer Science and Engineering Department, MES College of Engineering, Under the Calicut University, Kerala.

