IJCTA, 9(10), 2016, pp. 4285-4291

Implementation of LMMC and SBA Algorithms in Cloud Infrastructure Using Cloudsim

*S. Indirani **Dr. C. Jothi Venkateswaran

Abstract : In data centers Virtualization technology has improved server utilization and server consolidation, while reducing resource management complexity and costs. The important issue in cloud computing environment is the Live Migration in Virtual Machines. This paper, apply the combination of Seed Block Algorithm (SBA) and Location based Minimum Migration in Cloud (LMMC). It checks the number of tasks running already and a number of tasks able to run. LMMC monitors the Migration of a VM from overloaded tasks then to schedule another VM makes it possible to improve resource utilization for better load balancing, and avoid fault. SBA have been used to backup and recovery of failure tasksprocessed in VM.The main benefits of implementing fault tolerance in cloud computing include failure recovery, power consumption lower cost, load balancing, live migration safe scheduling, improved performance metrics etc.The results produced by the simulation illustrate the efficiency of LMMC and SBA approaches towards fault tolerance in terms of power savings, execution time and less cost.

Keywords : Cloud Infrastructure, Live Migration and Virtualization, Location based Minimum Migration in Cloud (LMMC), Seed Block Algorithm (SBA), CloudSim

1. INTRODUCTION

Cloud computing can be described as an internet based computing solution which is regarded as one forward step in the technological development of distributed computing. It yields an extensive solution for the delivery of IT in the form of a service and it enables scalable and cooperative sharing of resources between various organizations. The cloud facilitates on demand access to applications from any place in the world, without any consideration towards their details of implementation [1].

Cloud Computing reduce or avoid capital expenditure by renting third party services. Customers use resources as services and only pay for what they use. Like electricity and water services the customers pay only for their consumption. Cloud provides to its users, massive storage and maintenance of large volumes of data in reality. Workload is balanced efficiently, since the cloud can scale dynamically [2].

The main contribution of this paper is Migration of Virtual machine tasks where less number of tasks are running, which occupies extra energy and power consumption. The Migration of VM tasks where more number tasks are running, which gives overhead to the system performance. Less number of migrations should be selected, to avoid more processing cost for migration process.

This paper is organized as follows. Section II describes the related work. SectionIII presents methodology. Section IV presents problem formulation. Section V presents experimental results and Section VI presents conclusions and future work.

* Research Scholar, Mother Teresa Women's University, Kodaikanal-624 1022

** Head of the Department, Department of Computer Science, Presidency College, Chennai – 600005. jothivenkateswaran@yahoo.co.in

2. RELATED WORK

Even though load balancing, job scheduling, and fault tolerance are few of the active areas of research in cloud environments, these areas have highly been and are continued to develop independent of each another with everyone concentrating on several aspects of computing. Ghosh et al [4] presented two techniques referred to as deallocation and overloading, for improving scalability when rendering fault tolerance with a low overhead at the same time. Multiple backup copies in the overloading scheme may have an overlap in the same time slot on the same processor. The deallocation scheme is basically reclaiming of resources that are reserved for backup copies when the respective primary copies are successfully completed.

Next proposed is a fault tolerant and recovery system referred to as FRAS system (Fault Tolerant and recovery Agent System) [5]. This is an agent based system comprising of four kinds of agents. Recovery agent conduct roll back recovery after the failure occurs. Facilitator does the control of the communication between agents and garbage collection agent is responsible for the garbage collection of data. Agent recovery algorithm is introduced in order to keep up consistency in the state of a system and avoid domino effect.

The fundamental idea of live migration algorithm was first introduced by Clark et al [6]. At first, Hypervisor makes a mark over all pages as dirty. Then the algorithm transfer dirty pages across the network in an iterative manner till the number of pages that are remaining to be transferred is below a specific threshold or a maximum number of iterations is attained. Then Hypervisor does the mark on the transferred pages as clean. As the VM operates during live migration, hence the memory pages already transferred may be dirtied during iteration and require to be re-transferred. The VM is stopped at some point on the source for terminating further memory writes and then do the transfer of the remaining pages. The VM restarts at destination server, after transferring all the memory contents.

Beloglazov et al [7] investigated some heuristics for practical dynamic allocation of VMs for the improvement of the system resource utilization by employing live migration, and moving hosts that are idle to sleep mode.

DejaVu – is defined as a framework [8] which (1) reduces the resource management overhead by the identification of a small set of workload classes for the purpose of which it requires the evaluation of resource allocation decisions, (2) rapidly adapts itself to variations in the workload through the classification of the workloads making use of signatures and caching their desired resource allocations at runtime, and (3) tackles interference through the estimation of an "interference index".

The necessity of highly efficient fault tolerant scheduling for practical tasks in virtualized clouds has led to the attention being paid on virtualization and the elasticity and then FESTAL, *i.e*., Fault-tolerant Elastic Scheduling algorithms [9] was developed for real-time Tasks in clouds. A new fault-tolerant mechanism is developed that is an extension of the traditional PB model in order to include the features of clouds. As a first time, an elastic resource provisioning strategy is presented in the fault-tolerant scenario in order to optimize resource utilization when yielding support for fault tolerance in the case of clouds. Based on the fault–tolerant strategy and the elastic resource provisioning technique, a resourceful fault-tolerant scheduling algorithm–FESTAL is designed.

3. METHODOLOGY

The proposed method is the integration of Seed Block Algorithm (SBA) and Location based Minimum Migration in Cloud (LMMC) for the cause of Fault Tolerance and Efficient Scheduling in Cloud [3]. In order to provide services that are reliable, the fault-tolerant degree should be guaranteed when moving VMs for balancing loads.

The LMMC chiefly focuses on less migration in order to avoid the migration cost. The running virtual machines that are loaded to the minimum should not process migration. In fact, they require migration sometimes, heavy load in the virtual machines need live migration to one or more VMs from the closer location of cloud centers.

This integrated SBA and LMMC both for fault tolerance and efficient scheduling with migration in the cloud. When a new task arrives at the host, the task is first scheduled dynamically. Next the task model is processed and scheduling is done dynamically for every new task that arrives in the host.

The next subsequent step is the monitoring of the load balance of virtual machines if loaded heavy or light. Next when VM is needed for migration then it proceeds for live migration with lower cost for scheduling the task in VMs by making use of the LMMC algorithm

4. PROBLEM FORMULATION

The cloud in this work is specified by an infinite set $\mathrm{H} = \{h_1, h_2, ..., h_m\}$ of hosts with multiple processing power Pi which is measured in Million Instructions Per Second (MIPS), an extensively used metric [10-11], denoting the infinite heterogeneous computing resources present in the cloud. It is to be noted that not every host in the cloud is observed to be active; the count of active hosts is typically limited in reality. Hence the active hosts set is modeled by $\mathrm{H}_a = \{h_1, h_2, ...\ h_{j|\mathrm{H}_a}|\}$, $\mathrm{H}_a = \subseteq \mathrm{H}$, whereas the items in H "Ha indicate the hosts in sleep mode. The hosts can be switched dynamically between the active and the sleep mode on the basis of the system workload. Each host hi comprises of multiple VMs represented by a set $\mathrm{V}_i = \{v_{i1}, v_{i2}, ..., v_{ij} \mid v_i \mid\}$ of VMs having different processing power Pij . Further, the VMs can be dynamically generated and cancelled depending on the system. A set $\mathrm{T} = \{t_1, t_2, ...\ t_n\}$ of independent and non-preemptive tasks is used.

A. Fault Model

This work focuses over the hosts failure. In the case of a host failure, the tasks copies on this host will not go to complete. At one time, at the most one host might face a failure, in other words, in case the primaries performing the tasks fail, the backups can at all times complete them successfully before another host failure. Failures can be either temporary or permanent, and are autonomous, impacting only one single host. Additionally, there is also a failure detection mechanism like fail-signal and acceptance test that provides information about failures. There will not be any scheduling of new tasks to a host that failed. It is again to be noted that it seems pretty straight to have this model extended for tolerating multiple failures of host. First and foremost, the hosts are to be divided into many small groups. Then, the abovesaid fault model is applied to each small group. At last, the proposed fault tolerant mechanism is employed to each small group to deal with multiple host failures.

B. Scheduler Model

A scheduler can be deployed in a distributed or a centralized manner. Scheduling architecture follows the star topology communication model that is extensively analyzed in clouds [12] where the scheduler is held responsible for the cause of task scheduling to all the hosts, and thereafter monitoring the status of every host. The hosts present in the cloud provide report regarding their status info which is inclusive of the tasks' execution status to the scheduler straightly. On the arrival of a new task, the Seed Block Algorithm is called upon and scheduling is performed as soon as possible. Each task contains three attributes: arrival time ai, deadline di, and task size tsi which are measured by Millions of Instructions (MI) identical to [13-14]

5. EXPERIMENTAL RESULTS

Cloud computing is one of the foremost technologies for delivering robust, secure, fault-tolerant, sustainable, and scalable computational services. Cloud computing infrastructures and application services, allows its users to concentrate on particular system design issues that have to be investigated, without any concern over the details at the low level corresponding to Cloud-based infrastructures and services. It supports modeling and virtualized server hosts simulation, along with the policies which are customizable for preparing host resources for the use of virtual machines.

In this system CloudSim is used for rendering support for task backup, task scheduling, and provisioning of resource at the workflow level in the cloud. They are equipped with 5 VMs and each Virtual Machine (VM) has up to 10000 MB of memory and the capacity for processing up to 1,000 million instructions per second. The default network bandwidth is 15MB based on the execution environment in CloudSim, where the traces were collected.

In this experimental work, the network bandwidth indicates the maximum allowed data transfer speed between a pair of virtual machines per task. For the purpose of ensuring the experiments repeatability, and the simulation approach is employed for the evaluation of the performance of aforementioned algorithms. CloudSim [15] is selected for the simulation of the cloud data center in experiments. CloudSim, that supports the seamless modelling, simulation, and evaluation of large-scale cloud infrastructures, is a widely popular simulator in both industry and academics. The setting in detail and parameters that are utilized in the CloudSim are listed as below:

- Each host is modelled with a performance equivalent to 250, 500, 750,1000 MIPS;
- Four kinds of VMs having the processing power equivalent to 250, 500, 700 and 1000 MIPS are assumed;
- According to [16], the time needed for turning on a host and creating a VM is fixed as 90s and 15s, respectively.

First adding number of cloudlets or user task should be conducted in the CloudSim, and also the input size, output size, length of the tasks is added. The following Table 1 describes it.

Table 1. User Task

VMs	*MIPS*	*Image Size (MB)*	*RAM (MB)*	*Bandwidth (MB)*
0	250	10000	512	1000
1	500	10000	256	2000
2	750	10000	512	1000
3	1000	10000	1024	2000
4	500	10000	512	1000

Next adding Cloud Resources are performed as per following Table. 2.

Table 2.

Cloudlet ID	*Length (MB)*	*Input Size (MB)*	*Output Size (MB)*
0	10000	200	300
1	20000	400	600
2	10000	600	300
3	20000	400	600
4	40000	600	900
5	10000	200	300
6	20000	400	600
7	10000	600	300
8	20000	400	600
9	40000	600	900

In this workflow, cloud resources having virtual machines with each having capacity of performing the task, RAM size and MIPS are assigned and cloudlet must have the range of input size, cloudlet ID dynamically. The improvement attained by the fault tolerant methods is less comparable to the performance improvements with respect to power consumption and load balancing. After scheduling user tasks 200 is allocated to 1 and VM 1 run only one task performs power consumption is 100 joules. The task 400, id1, is allocated to 0 and VM 0, 1 run the task each 1 and consume power each 100 joules. The task 600 id2 is allocated 0 and VM 0, 1 run 3 tasks. And then the task 400,600,200,400, 600,400,600 are allocating to the VM 0,1 and number of task is 4,5,6,7,8,9,10 respectively. Hence the tasks are scheduled every time so that power consumption becomes high. After scheduling, the power that is consumed by the VM is 300 joules. The migration cost also increases slightly and virtual machine load balancing gets better in this migration.

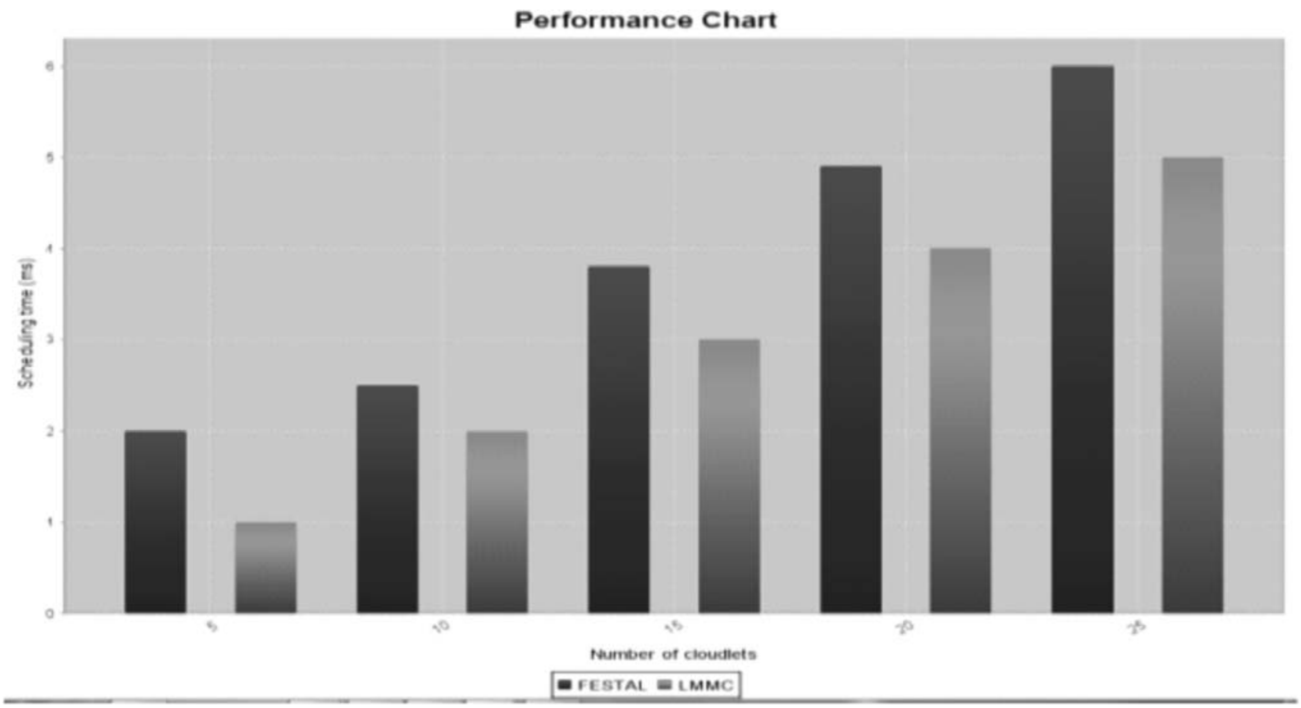

Fig. 1. Scheduling time comparison .

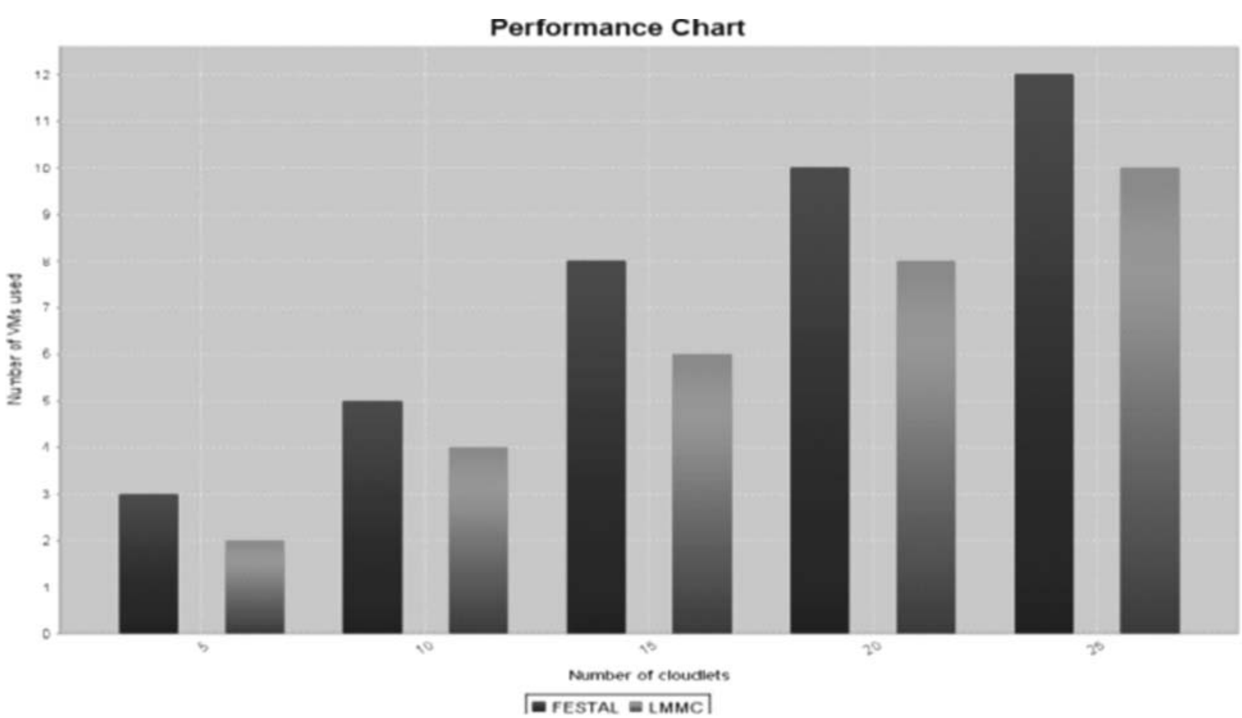

Fig . 2. Cloud cost comparison.

After LMMC, the simulation results indicate the task performance and migration of task with load balancing gets efficient speed and cost is also reduced in that scheduling. Before scheduling the VM 0, 1, 2 are employed for running the task. After LMMC VM 1 migrated the task to VM 0 and VM 0 is run with 7 tasks, VM 1 runs 3 tasks and also the power consumed is 200 joules. The performance of cost, scheduling, VM usage is compared with FESTAL which are shown in Fig.1, Fig.2 and Fig.3. Scheduling graph of utilizing LMMC is plotted through the given tasks which is shown in Fig.1. This result illustrates that dynamic scheduling of the tasks in comparison with Festal. In Fig.2 cost graph of LMMC is drawn for the given tasks. The result specifies the cloud costs that are used in the particular number of tasks in this system. In Fig.3 VMs usage graph is drawn by making use of the given tasks. This result indicates the number of virtual machines which is used in specific number of tasks used in the LMMC in comparison to Festal. It can be concluded from the above that LMMC is better in comparison to the Festal since its power consumption is less with low cost and VMs that are used in scheduling for migration in that cloud are compared with Festal.

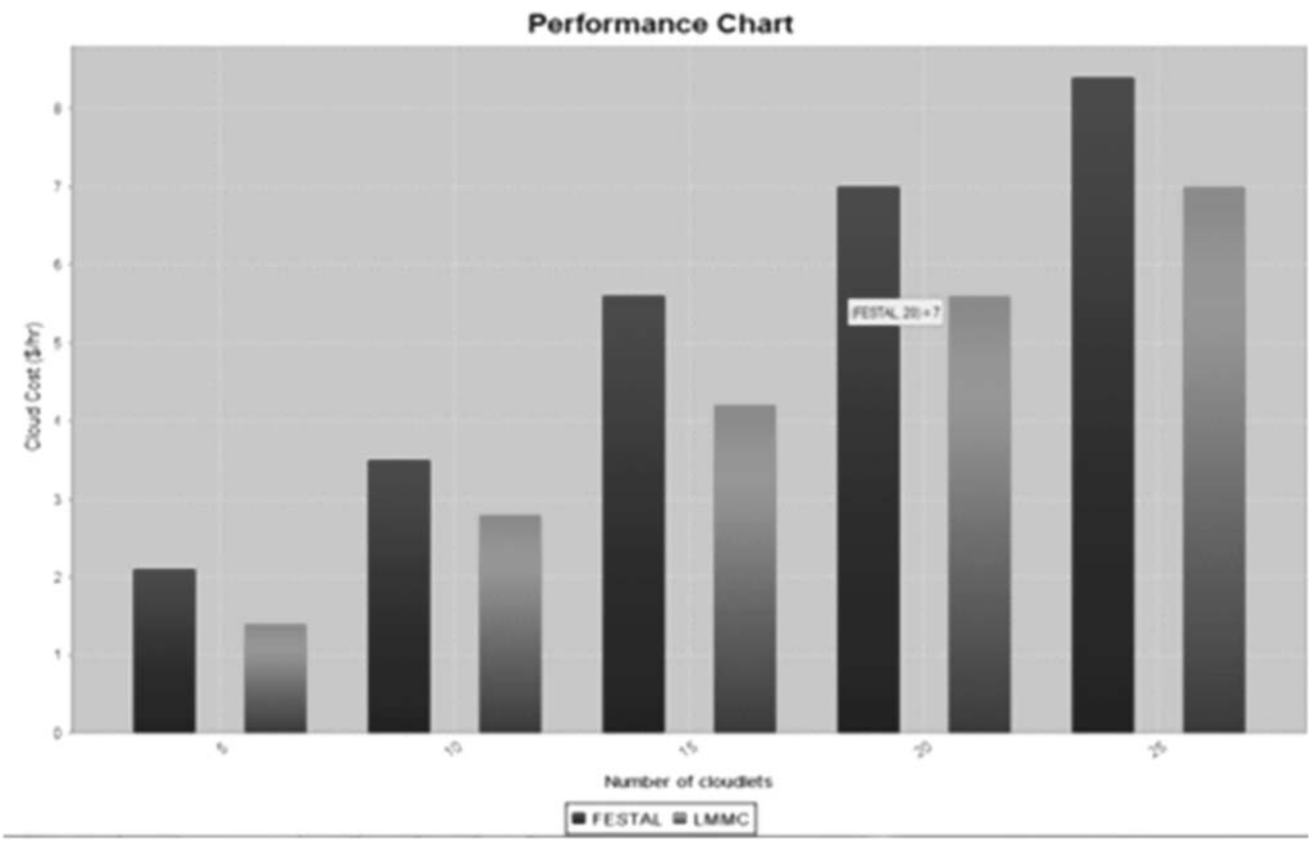

Fig. 3. VMs usage model.

6. CONCLUSION AND FUTURE WORK

In this paper, significant focus is on the load balancing algorithm based on fault tolerance policy. They are used with the combination of Seed Block Algorithm (SBA) and Location based Minimum Migration in Cloud (LMMC) for Fault Tolerance and effective Scheduling in Cloud. The flexible backup task is the number of faults and recovery of the task reprocess. When new task arrives at VM and if it is considered overloaded then, then LMMC can improve both resource utilization and task response time while simultaneously avoiding a situation whence some VMs are loaded heavily with others being idle or performing little work and then dynamically scheduling the task and improving the load balance. These techniques for efficient live virtual machine migration helps in reducing both recover time and total migration time. Experimental results indicate the power consumption in the entire process of migration of the VMs and less cost for fault tolerant in cloud. On a futuristic note, other kinds of fault-tolerance like data corruption could be examined. Further the experiments done in this research were time limited and resource constrained.

7. REFERENCES

1. P. Mell, T. Grance, The NIST Definition of Cloud Computing, NIST Special Publication 800-145, 2011.
2. Ravichandran, S., Naganathan, E.R., (2013) "Dynamic Scheduling of Data Using Genetic Algorithm in Cloud Computing", International Journal of Computing Algorithm, 2(1), 127.
3. Indirani, S., & Jothi Venkateswaran, C., (2016) "Live Migration Approach of Fault Tolerance in Cloud Infrastructure using LMMC and SBA Algorithms"
4. Ghosh, S., Melhem, R., & Moss´e, D., (1997) "Fault-tolerance through scheduling of aperiodic tasks in hard real-time multiprocessor systems" , IEEE Transactions on Parallel and Distributed Systems, 8(3), 272–284.
5. Lee, HM., Chung, KS., Chin, SH., Lee, JH., Lee, DW., (2005)"A Resource Management and Fault Tolerance Services in Grid Computing", Journal of Parallel and Distributed Computing, 65, 1305-1317.
6. Christopher, C., Gorm, HJ., Steven, H., Eric, J., Andrew, W., Ian, P., & Christian, L., (2005) "Live virtual machines migration," In Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation, USENIX Association, 2, pp. 273-286.
7. Beloglazov, A., Abawajy, J., &Buyya, R., (2012) "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing", Future generation computer systems, 28(5), 755-768.
8. Vasiæ, N., Novakoviæ, D., Miuèin, S., Kostiæ, D. and Bianchini, R., (2012) "Dejavu: accelerating resource allocation in virtualized environments" In ACM SIGARCH Computer Architecture News, 40(1), 423-436.
9. Wang, J., Bao, W., Zhu, X., Yang, L. T., & Xiang, Y., (2015) "FESTAL: Fault-Tolerant Elastic Scheduling Algorithm for Real-Time Tasks in Virtualized Clouds", IEEE Transactions on Computers, 64(9), 2545-2558.
10. Xu, B., Zhao, CE., Hu, & Hu, B., (2011) "Job scheduling algorithm based on berger model in cloud environment," Advances in Engineering Software, 42(7), 419–425.
11. Wu, L., Garg, SK., & Buyya, R., (2012) "Sla-based admission control for a software-as-a-service provider in cloud computing environments," Journal of Computer and System Sciences, 78(5), 1280–1299.
12. Sadhasivam, S., Nagaveni, N., Jayarani, R., & Ram RV., (2009) "Design and implementation of an efficient two-level scheduler for cloud computing environment," In IEEE International Conference on Advances in Recent Technologies in Communication and Computing, 884– 886.
13. Tsai, M-Y., Chiang, P-F., Chang, Y-J., & W.-J. Wang, (2011) "Heuristic scheduling strategies for linear-dependent and independent jobs on heterogeneous grids", Springer.
14. Haselhorst, K., Schmidt, M., Schwarzkopf, R., Fallenbeck, N., & Freisleben, B., (2011), "Efficient storage synchronization for live migration in cloud infrastructures", In 19th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), 511-518.
15. Calheiros, RNR., Ranjan, A., Beloglazov, CA., De Rose, & Buyya, R., (2011) "Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," Software: Practice and Experience, 41(1), 23–50.
16. Hermenier, F., Lorca, X., J.-M. Menaud, G. Muller, and J. Lawall, (2009) "Entropy: a consolidation manager for clusters," In ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments, Conference Proceedings, 41–5.
17. 0.