Enhanced Resource Scheduling Approach for Cloud Computing.

E.O. Oyebode, MSc.[1*]; O.A. Ojesanmi, Ph.D.[2]; O. Folorunso, Ph.D.[3]; and I.K. Ogundoyin, Ph.D.[4]

[1]Department of Computer Science, Ajayi Crowther University, Oyo, Oyo State, Nigeria.
[2]Department of Computer Science, Federal University of Agriculture, Abeokuta, Ogun State, Nigeria.
[3]Department of Computer Science, Federal University of Agriculture, Abeokuta, Ogun State, Nigeria.
[4]Department of Information and Communication Technology, Osogbo, Osun State, Nigeria.

E-mail: kunlebest4u@yahoo.com*

ABSTRACT

The various arrangements of resources in terms of assigning hosts to virtual machine, placing tasks on virtual machine and how each virtualized resource is released to the tasks can be handled efficiently via scheduling. Considering task dependency of workflows in parallel execution in cloud scheduling of resources can improve resource utilization and throughput in cloud computing. In this study, various task execution scenarios have been examined and identifying the dependencies of task before allocating resources in both time sharing and space sharing policies have shown improvement in resource utilization and throughput.

(Keywords: host, virtual machine, workflows, scheduling, time sharing, space sharing)

INTRODUCTION

The need to have more resource for sharing on different platforms has led to cloud computing. Resource pool is established and managed inform of data center. The resource sharing platforms can be used for service delivery via the internet. As a result of this development, an increasing amount of computation is now hosted in private and public clouds. At the same time, datacenter resource efficiency (i.e., the effective utility from the resource pool needs to increase). However, utilization of resource from the cloud pool of resource has been very low, rarely exceeding 20-30%, (Delimitrou *et al.,* 2015). Low utilization coupled with the lack of scaling in hardware due to technological limitations pose threatening scalability roadblocks for cloud computing, (Gandhi, 2013).

There is need to adopt techniques that can increase the compute capabilities of datacenters. Various solutions have been proposed such as cloud federation. Solutions range from increasing the number of servers in each datacenter to cloud federation that allows two or more clouds to integrate, (Grozev and Buyya, 2014). Increasing the number of servers also automatically leads to increase cost in power and cooling maintenance and load balancing issues. Cloud federation could extend data center resource capacity but has increased cost and security challenges (Toosi, *et al.,* 2014).

The focus of this work is to examine various strategies of utilizing the cloud resources with respect to workflows, task dependencies, servers, host capacities and to propose hybrid strategy for scheduling tasks and cloud resources for improved resource utilization.

LITERATURE REVIEW

The various arrangements of resources in terms of assigning hosts to virtual machine, placing tasks on virtual machine and how each virtualized resource is released to the tasks can be handled efficiently via scheduling. To create infrastructure from resources, the user specifies the instance type and the VM image as supported by the cloud provider, after the VM image has been transparently deployed on a physical machine (the resource status will begin to run), after which the instance is booted and in the process, the resource status becomes operational. However, there is limitation to maximum number of instances that can be used together concurrently, (Hwang and Fox, 2012).

The amount of hardware resources available to VM during instantiation request is constrained by the total processing power and system bandwidth available within the host. Failure to consider the host resource level can lead to over provisioning of VMs that can affect the tasks meant for execution on the VM and cause service level objective (SLO) violations.

Existing techniques for sharing resources in the cloud include Round-Robin scheduling which focuses on distributing the tasks equally among the virtual machines following a fixed order among the servers (Agarwal and Jain, 2014). The main advantage of this algorithm is that it utilizes all the resources in a balanced order. Round-Robin approach is good when all the tasks and virtual machines have the related capacities. This is not true in the real world as tasks and machines may exercise different sizes and capacities with request patterns unpredictable. Power consumption, delay, reduced throughput can result from resource usage via round robin scheduling. Some of the efforts made to improve Round-Robin include the match-making algorithm.

The match-making algorithm applies filtering out of virtual machines that do not meet the tasks requirements of resources such as cpu, memory, processors etc, (Toosi *et al*., 2014). In other to distinguish different machine attributes, ranking was applied. OpenNebula implements rank scheduling policy that first allocates VMs to tasks on first come first serve basis. Gandhi *et al.* (2013) identified the three important stages of resource scheduling process as resource discovering, filtering and resource selection and then proposed priority algorithm that depends on user defined features to setup resource selection.

Singh *et al.* (2014) carried out performance evaluation of some algorithms used for scheduling. Kumar and Ravichandran (2012) used a classification method to classify tasks into two major groups using deadline and cost based. The work then applied FIFO queue and priority based queues.

Kaur and Kaur (2016) considered a model of resource scheduling with respect to heterogeneous cloud environment and proposed a hybrid genetic algorithm approach with a 2 stage algorithm approach and a case library. The user request types which have various attributes were modeled with directed acyclic graph (DAG) and considering dependencies among tasks. The tasks were identified and ranked according to resource needed. The approach assumed that the load information among the processors cannot be determined. Most works considered did not examine the feature of parallel execution of submitted tasks.

It is necessary to consider the important features of the cloud with respect to resource sharing. Workloads and task execution strategies can be used as basis for scheduling. Workloads can exhibit different patterns. Workloads in grid environment are characterized by small bags-of-tasks and workflows with sequential tasks. Most works on scheduling of resource in the cloud do not consider multiple release of resources to tasks. For instance, scientific computing tasks require multiple resource release. Flexiscale is one of the cloud providers, queuing list of Flexiscale can exceed two weeks whenever overload is experienced (Hwang *et al.,* 2011).

Most works on scheduling of resource in the cloud do not consider multiple release of resources to tasks. Many task computing workloads involve the use of loosely coupled applications to achieve their goals. Such workloads may consist of tens of thousands to hundreds of tasks and bag-of-tasks (BoTs).

From recent reviews, identifying this type of users can be through identification of users with many submitted tasks or bags-of-tasks in the workload traces. Such traces are common in real scientific traces and can be allocated by considering the volume of tasks and interval of time within submission, (Taoshen and Zhang, 2014). For instance, using internal ip addresses, i.e ip address accessible only within a cloud can optimize data transfers between closely-located instances as it reduces traffic congestion and operational tasks, improve the resource utilization and also improve the server performance and throughput.

Some of the parallel applications show a decrease in utilization of CPU resources whenever there is an increase in parallelism if the jobs are not scheduled correctly then it reduces the computer performance. Basically, workflow has a set of tasks which may be more or less dependent. Submission policy for execution of multiple tasks in the cloud can be done through space-share policy and time-share policy. In cloud computing, space-share and time-share policies are used to allocate resource to tasks,

(Calheiros *et al.,* 2011). In a space-share environment, the scheduler assigns specific CPU cores to specific tasks. Space-share scheduling can be likened to first come first serve sequence.

A task gains access to resource usage and continue to execute until the task is completed, after completion, another task awaiting execution then receives attention. This principle is also being applied to hosting of multi[ple virtual machines on different locations to determine the number of VMs that can run concurrently. Time space sharing scheduling policy is a job scheduling policy that allows multi-process execution of tasks via allocation of time to execution of tasks. All the available jobs within a time interval are submitted to virtual machine for execution at once. The virtual machine then allows multiple execution of tasks.

PROPOSED RESOURCE SHARING APPROACH

In other to measure the efficiencies of the two main scheduling policies, various experiments were setup using CloudSim. CloudSim was developed in Java for the purpose of modeling and simulation of the cloud, (Buyya *et al.,* 2009). It is a layered structure with architecture comparable to modern day cloud. It includes library that is sufficient for testing various algorithms proposed for the cloud. CloudSim allows instantiation and execution of entities such as VMs, cloudlets, datacenters, applications etc. CloudSim allows VM provisioning at both the host level and VM level.

At the host level, it allows specification of how much of the overall processing power of each core will be assigned to each VM during instantiation. And at the level of VM, a fixed amount of the available processing power is released to the individual application services (tasks units) that are hosted within the VM. Other tasks assigned are kept waiting in queue for execution within the VM. In this work, we define the scheduling as having three tuples: resources, mapping of resources with task and total execution interval

Schedule = (R,M,TEI) **(1)**

Scheduling depends on available cloud resources, where resources R can be expressed as:

$R = \{r_1, r_2, r_3, \ldots, r_n\}$ **(2)**

Each resource r_i has a VM type VM_{ri} associated with it.

$$m_{t_i}^{r_j} = (t_i, r_j, \ldots) \quad \textbf{(3)}$$

M represents a mapping that was used to assign tasks, t to resources, j (virtual machine) and is comprised of tuples.

Rodriguez and Buya (2014) showed that for every VM type, the processing capacity can be estimated and base on policy for execution, Equations 4 and 6 were used for estimation of task execution of time on VM.

By using a space-shared policy, the estimated finish time of a task managed by a VM *i* is given by:

$$Tct = st + \frac{\sum_{i=1}^{n} inst}{cap * cores(t)} \quad \textbf{(4)}$$

p represents task, st is start time of a task, inst is the number of instructions required for execution of tasks, cores(t) is the number of cores required by the cloudlet.

$$\text{Where cap} = \frac{\sum_{i=1}^{p} cap(vm)}{p} \quad \textbf{(5)}$$

In a time-shared environment task scheduler will distribute the capacity of core among different tasks dynamically.

$$\text{Task\_completion\_time} = sm + \frac{\sum_{i=1}^{n} inst}{cap * cores(t)} \quad \textbf{(6)}$$

$$\text{Host\_Capacity} = \frac{\sum_{i=1}^{p} cap(vm)}{max(\sum_{c=1}^{cloudlets} cores(c), p} \quad \textbf{(7)}$$

For task dependency, subtask:

$S = \{s_1, s_2, s_3, \ldots, s_n\}$, **(8)**

where s_i is ith subclass to be decomposed into

(i=1, 2,…,n) **(9)**

and n represents the number of subtasks. Direct acyclic graph (DAG) was used to express the dependency among tasks where:

DAG = (V, E) **(10)**

where V represents the number of tasks as nodes and dependency among the tasks as edges, E. The edges were ranked and used as priority for scheduling. The initial algorithm used for distribution of task is shown in Figure1.

```
Input Nos_of_Cloudlets, Nos_of_VMs
For all cloudlets, obtain filesize, SLO
For all vms, compute MIPS
If (nos_of_Cloudlets <= Nos_of_VMs) {
        Sort cloudlet_file_size in ascending order
        Sort vm in ascending order
        For each v∈ Vms do
                Assign timesharing on v
                Submit cloudlets[i] and v[i] to host
        end
Else
        For each ci∈ C  do
                compute priority of Cloudlets
        end
        Sort cloudlets ci for all c in descending
order
While cloudlets <= vms
Assign spacesharing on v
Submit task cloudlets[i] and v[i] to host
        Repeat step until task is exhausted.
```

Figure 1: Task Distribution Algorithm.

RESULTS AND DISCUSSION

Figure 2 is the graph of the space sharing scheduling policy and time sharing scheduling policy using the same number of virtual machines and independent tasks. At initial stage, the two scheduling policies gave similar results when the number of tasks submitted for execution was far less than the number of virtual machine available in the data center. As the number of tasks gradually rises beyond the number of virtual machines available at the center, there is variation in the number of tasks executed with space sharing policy accomplishing more tasks at lower time than time sharing policy.

The performance of the two scheduling algorithms were further checked under dependent tasks. The plot is shown in Figure 3.

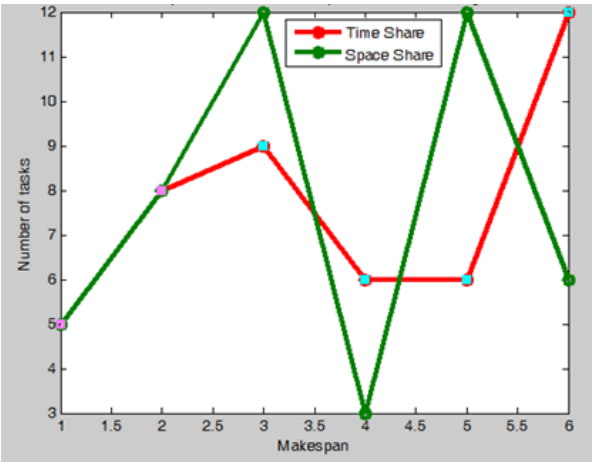

Figure 2: Space Sharing and Time Sharing Policy.

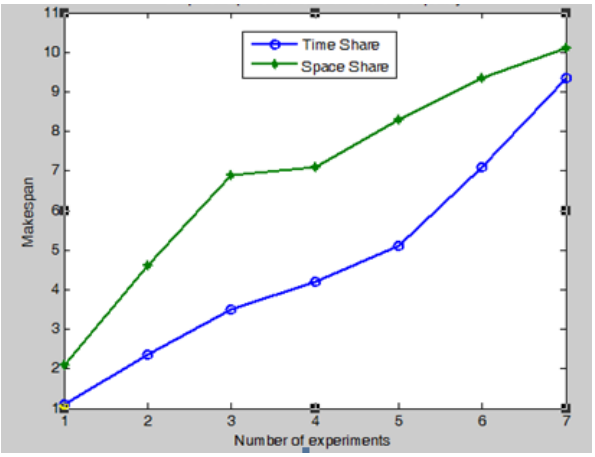

Figure 3: Space Sharing and Time Sharing Policy with Dependent Tasks.

A technique of task properties that examines the task requirements before allocation was considered. The tasks requirements were examined to decide ahead the likely task completion time. The unused VM in a time interval was used as available resource and then schedule tasks with respect to available resource and deadline attached. To further test the scheduling policy, the task dependency was used as a criterion for scheduling resource for task execution. The dependency ratio among tasks was used as priority such that tasks with high dependency ratio were schedule on VM that could execute the tasks. The performance of the two sharing policies is shown in Figure 4. Time

space sharing shows significant improvement in terms of task execution.

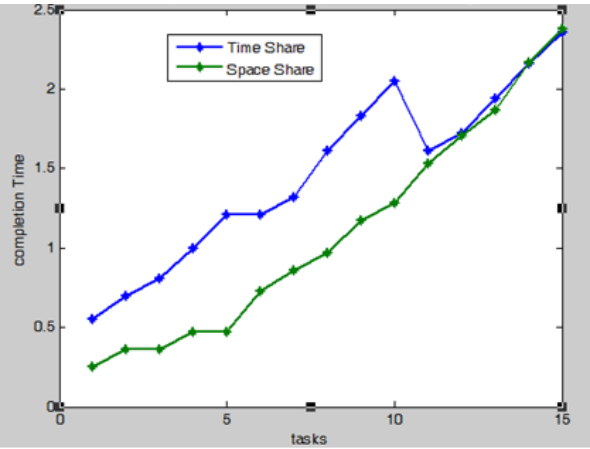

Figure 4: Space Sharing and Time Sharing Policy with Task Dependency and Deadline.

A measurement of resource utilization under consideration of task dependency and deadline is shown in Figure 5.

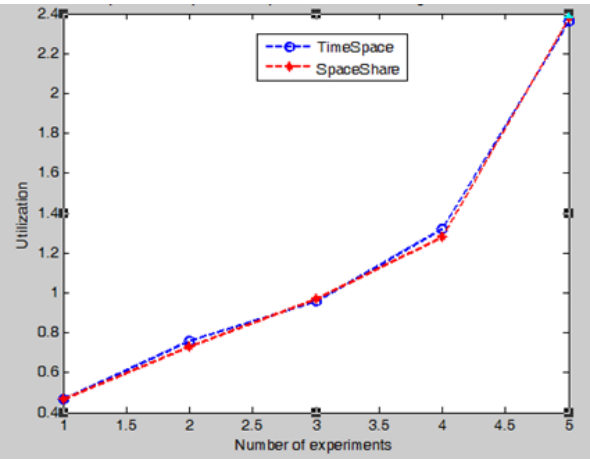

Figure 5: Resource Utilization.

CONCLUSION

This study presents a new scheduling scheme that considers the properties of the cloud in terms of parallel execution of tasks and dependency. The results show improvement in the utilization of cloud resource when dependencies among the tasks were considered. The overall implication is that the proposed approach can boost the resource utilization of cloud computing.

REFERENCES

1. Agarwal and Jain. 2014. “Efficient Optimal Algorithm for Task Scheduling in Cloud Computing Environment”. *International Journal of Computer Trends and Technology*. 7(9):344 –349.

2. Buyya, R., R. Ranjan, and R.N. Calheiros. 2009. “Modeling and Simulation of Scalable Cloud Computing Environments and the Cloudsim Toolkit: Challenges and Opportunities”. *Proceedings of the 2009 International Conference on High Performance Computing and Simulation*, HPCS 2009. 1–11.

3. Calheiros, R.N., R. Ranjan, A. Beloglazov, A.F. C´esar, C.A.F. De Rose, and R. Buyya. 2011. “CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms”. *Software. Practice and Experience*. 41:23–50.

4. Delimitrou, C., G. Hall, and S. Mall. 2015. “Paragon: QoS-Aware Scheduling for Heterogeneous Datacenters”. Eighteenth International Conference on Architectural Support for Programming Languages and Operating Systems, (ASPLOS).

5. Gandhi, A. 2013. “Dynamic Server Provisioning for Data Center Power Management”. (Unpublished Doctoral dissertation). Computer Science Department School of Computer Science Carnegie Mellon University: Philadelphia, PA.

6. Grozev, N. and R. Buyya. 2014. “Multi-Cloud Provisioning and Load Distribution for Three-Tier Applications”. *ACM Transactions on Autonomous and Adaptive Systems*. 9(3):1–21.

7. Hwang, K., G. Fox, and J. Dongarra. 2011. *Distributed and Cloud Computing: From Parallel Processing to the Internet of Things*. Morgan Kauffmann Publishers: London, UK.

8. Kaur G. and S. Kaur. 2016. “Improved Hyper-Heuristic Scheduling with Load-Balancing and RASA for Cloud Computing Systems”. *International Journal of Grid and Distributed Computing*. 9(1):13--24

9. Kumar, B.A. and T. Ravichandran. 2012. “Time and Cost Optimization Algorithm for Scheduling Multiple Workflows in Hybrid Clouds”. *European Journal of Scientific Research*. 89(2):265-275.

10. Singh, R.M., S. Paul, and A. Kumar. 2014. “Task Scheduling in Cloud Computing: Review”. *International Journal of Computer Science and Information Technologies*. 5(6):7940-7944.

11. Tao, F., Y. Feng, L. Zhang, and T.W. Liao. 2014. "CLPS-GA: A Case Library and Pareto Solution-Based Hybrid Genetic Algorithm for Energy-Aware Cloud Service Scheduling". *Applied Soft Computing Journal*. 1–16.

12. Taoshen, L. and X. Zhang. 2014. "On the Scheduling Algorithm for Adapting to Dynamic Changes of User Task in Cloud Computing Environment". *International Journal of Grid Distribution Computing*. 7(3):31-40.

13. Toosi, A. N., R.N. Calheiros, and R. Buyya. 2014. "Interconnected Cloud Computing Environments". *ACM Computing Surveys*. 47(212):1–47.

ABOUT THE AUTHORS

Oyebode Ebenezer is a Lecturer at the Ajayi Crowther University, Oyo. He holds a Master of Science (M.Sc.) in Computer Science from Obafemi Awolowo University. He is a Ph.D. student in Computer Science at Federal University of Agriculture, Abeokuta. His research interests are in cloud computing, database management, mobile computing and artificial intelligence.

Dr. Ojesanmi Segun is a Senior Lecturer in the Department of Computer Science at Federal University of Agriculture Abeokuta. He holds a Ph.D. degree in Computer Science. He is currently serving as the Head of Department, his research interests are in the areas of mobile computing.

Dr. Folorunso Olusegun is an Associate Professor in the Department of Computer Science at Federal University of Agriculture Abeokuta. He is currently serving as the Director of Webometrics for the Federal, his research interests are in the areas of artificial intelligence, expert system, informatics, and database management system.

SUGGESTED CITATION

Oyebode, E.O., O.A. Ojesanmi, O. Folorunso, and I.K. Ogundoyin. 2017. "Enhanced Resource Scheduling Approach for Cloud Computing". *Pacific Journal of Science and Technology*. 18(1):146-151.