



# Efficient Cloudlet Allocation to Virtual Machine to Impact Cloud System Performance

Lizia Sahkhar, National Institute of Technology Meghalaya, India\*

Bunil Kumar Balabantaray, National Institute of Technology Meghalaya, India

 <https://orcid.org/0000-0002-2769-7122>

Satyendra Singh Yadav, National Institute of Technology Meghalaya, India

 <https://orcid.org/0000-0002-7891-6997>

## ABSTRACT

Performance is an essential characteristic of any cloud computing system. It can be enhanced through parallel computing, scheduling, and load balancing. This work evaluates the connection between the response time (RT) and virtual machine's (VM's) CPU utilization when cloudlets are allocated from the datacenter broker to VM. To accentuate the RT and VM's CPU utilization, a set of 100 and 500 heterogeneous cloudlets are analyzed under hybridized provisioning, scheduling, and allocation algorithm using CloudSim simulator. These include space shared (SS) and time shared (TS) provisioning policy, shortest job first (SJF), first come first search (FCFS), round robin (RR), and a novel length-wise allocation (LwA) algorithm. The experimental analysis shows that the RT is the least when SJF is combined with RR allocation at 40.665 seconds, and VM's CPU utilization is the least when SJF is combined with LwA policy at 12.48 in all combinations of SS and TS provisioning policy.

## KEYWORDS

Cloud Computing, CloudSim Simulator, First Come First Search, Resource Allocation, Response Time, Round Robin, Shortest Job First, Time Shared and Space Shared Scheduling Policy, VM's CPU Utilization

## INTRODUCTION

Cloud computing paradigm is an internet-based model composed of an extensible and scalable computing entity which requires minimal management effort and service provider interaction (Mell & Grance, 2011). Cloud computing promotes availability, scalability, reliability and portability in a computing system (Siegel & Perdue, 2012). The anytime-anywhere access policy of resources provided by cloud computing environment combined with offered storage capacity has improved the quality of service for the end-users to a great extend (Khaire, 2017). Likewise, cloud computing

DOI: 10.4018/IJISMD.297630

\*Corresponding Author

This article published as an Open Access article distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/4.0/>) which permits unrestricted use, distribution, and production in any medium, provided the author of the original work and original publication source are properly credited.

improves market and enterprises by reducing initial investment and capital expenditure promoting industrial specialization and resource utilization (Khaire, 2017).

Cloud computing makes the notion of “pay for what you use” or “infinite availability” that is use as much as you want. Such kind of a service is applicable only when the backend of a system is robust, proficient and flexible. One of the factors that drive the efficiency of a cloud computing backend system can be seen in its virtualized environment (Kumar & Charu, 2015). The virtualized infrastructure of cloud computing offers a virtual version of operating system, server, storage and network resources to the cloud users thereby increases efficiency, throughput and overall cost-effective (Vaezi & Zhang, 2017). Microsoft Azure and Amazon web services (AWS) are the most popular and scalable cloud computing services (Kotas et al., 2018) offers to cloud users. Therefore, even though different, cloud computing and virtualization share a close connection and a common bond. To sustain service to cloud users, cloud service provider has to maintain the standard of quality service without fail. For efficient performance, the main concern of any cloud computing system is balancing the workload shared among its virtualized component. Proper scheduling and load balancing improves response time, processing time, resource utilization, overall execution time, throughput, scalability and associated overheads (Deepa & Cheelu, 2017). Enhancing scheduling and balancing technique improves quality of cloud service and is an area which attracts the attention of many development and research work throughout the globe.

Performance is an essential property of any cloud computing system. It determines the functional efficacy and necessary improvements to harness the system performance (Jacob & Raj, 2019). Cloud performance can be achieved through parallel computing, load balancing and job scheduling (Khaire, 2017) and should be complete, efficient and guaranteed. This paper delves into the ambit of scheduling, load allocation and balancing techniques. Scheduling in cloud computing is a set of policies to regulate which task of the computer system would be taken up. Load balancing technique, on the other hand, is a process in which no node of a system remains in idle state while others are over utilized. Load balancing can be static and dynamic in nature. In static load balancing, prior knowledge about the node’s specification such as memory, bandwidth or processing elements, is required, and using this information, the load distribution is distributed accordingly at compile time (Deepa & Cheelu, 2017). Whereas, in dynamic load balancing, the load distribution happens at run time and in this case, decisions to distribute the load between the heavy and a lighter node happen dynamically (Deepa & Cheelu, 2017). Load balancing properties help to achieved prioritization and efficient allocation of resources thereby contribute to cloud service and performance.

This paper assessed the performance of a cloud computing system with respect to response time (RT) and VM’s CPU utilization. The analysis is done on a set of one hundred (100) and five hundred (500) heterogeneous cloudlets when the cloudlets are allocated from the data centre to the virtual machine using CloudSim simulator. The performance is evaluated by incorporating a hybridization of scheduling, allocation and load balancing algorithm under various environmental setups. The major contributions of this work are as follows:

1. To appraise a connection between RT and VM’s CPU utilization by each task request or cloudlet when the length or the instruction size of the cloudlet are arranged in ascending order of its length or when they are allocate in first come first serve basis.
2. Evaluate latency and utilization through RT and VM’s CPU utilization by each cloudlet under various environmental combinations of scheduling, load balancing and allocation technique. These includes space-shared (SS) and time-shared (TS) provisioning policy, first come first serve (FCFS), shortest job first (SJF), round robin (RR) and a novel length-wise allocation (LwA) algorithm.
3. Hybridized the evaluation technique such that each techniques takes into account the combination of SS and TS provisioning policy at the cloudlet and host level, FCFS or SJF algorithm to schedule the workload and finally with RR or LwA policy to allocate the cloudlets from the datacenter broker to the VM.

Following this introduction, the rest of this paper is organized as follow. Section 2 reviewed the related works and development in the subject. The scheduling policy of the CloudSim simulator is described in section 3. Section 4, present the objectives of this work, methodology and algorithm of the cloudlet allocation policy. The simulation setup and experimental evaluation of the proposed algorithms is presented under section 5 and finally section 6 concludes this paper.

## BACKGROUND

The flexible nature of cloud computing has increase the demand for its service tremendously. This lead to the mushrooming of Cloud service providers (CSP) which place cloud users in dilemma to choose CSP that satisfy the quality of service without fail. Saha et al. (2021) introduces a hybrid multi-criteria decision-making (HMCDM) model which helps cloud users to avoid such conflict by improving decision making and select the best CSP or alternative. HMCDM uses both analytic network process (ANP) and VIseKriterijumska Optimizacija I Kompromisno Resenje (VIKOR) to improve best decision making model among cloud users.

Hlaing and Yee (2019) uses CloudSim toolkit to allocate the incoming task requests from the data center broker to the VM in a cloud computing environment using a static independent task scheduling on virtualized servers. The allocation is based on resource availability such as processing power, processing elements and cost among others. The simulation minimizes execution cost and maximizes total execution time as compare to SJF and FCFS algorithm.

Shi et al. (2021) uses Bsufferage algorithm to analyze the performance of cloud task scheduling using CloudSim package. Bsufferage demonstrate improvememt in their completion time, throughput and load balancing of resources as compared to classical uffrage problem.

Roy et al. (2017) focus on finding the best cloudlet allocation technique to VM. They proposed a three-phase cloudlet allocation algorithm known as range wise busy checking 2-way balanced (RB2B). In this case, the advanced datacentre broker (ADCB) select a type of virtual machine for the cloudlet based on the cloudlet length and the range accepted by the VM. If no suitable VM is found, the ADCB will allocate the cloudlet to a local queue of the VM with the earliest finish time that satisfies local queue length limitation and balance threshold. On analysis, this scheme outperform state-of-the-art scheduling in term of waiting time and turnaround time besides other parameters.

Chien et al. in (2016) evaluate the average RT and processing time in TS and SS scheduling cases. The RT is based on the capacity, which is defined as the average processing power of the core elements of the VM, and the execution finish time. Upon evaluation, this work finds that the average RT and processing time is at its best when the cloudlet scheduler is TS and worst when it is SS.

Sujana et al. in (2020) work on minimizing the makespan of the schedule to achieve optimal scheduling techniques. This technique introduces a cost prediction based optimal secured scheduling algorithm to allocate the tasks to the most likely VM with security coverage. The algorithm combines heuristic cost prediction matrix and the fuzzy-based decision model to decide on the VM selection. WorkflowSim toolkit, an extension of CloudSim, is used to simulate the experiment and the result thus found yields efficient VM secured scheduling with comparable time complexity.

Kumar and Silambarasan (2019) uses MATLAB and Cloudsim to enhance the performance of cloud parameters such as CPU utilization, execution of user's request and remote storage of patient's digital information in an IOT healthcare setup. This is done by optimizing the resources of the VMs in cloud environment through optimization techniques such as particle swarm optimization, artificial bee colony optimization and cuckoo search algorithm. Using these techniques, the execution time has improved and become more efficient to process and handle real-time requests for the benefit of healthcare in the future.

Himthani and Dubey (2019) analyze the performance of various cloud parameters in TS and SS provisioning policy at the host and VM level. In their observation, waiting time is more in TS rather than SS scheduling policy. This is because in SS scheduling, if a cloudlet is assigned to a VM for

execution, VM will dispatch that cloudlet only after executing it completely whereas in case of TS scheduling multiple cloudlets execute over a single VM and hence waiting time is more. They also encounter that the datacentre debt is less in SS and higher side in TS environments. This is because in case of SS's VM scheduling, the numbers of VM initialized are less in number and the resources are inadequate comparatively. While in case of TS's VM scheduling, numbers of VM initialized are more as compared to SS scheduling and hence the datacentre debt is higher because more utilization of resources has been made by the executing VM.

Executing cloudlets on VM require high computational resource and hence cannot scale well. Recent advances in serverless computing make them appropriate for executing workloads on cloudlet. Nithya et al. (2020) introduces a software-defined cyber foraging framework (SDCF) for executing workloads on cloudlets by designing a light-weight wasm runtime for enabling serverless functions on Web-assembly that can be executed at multiple points or almost anywhere. Base on this study, resource allocation and scheduling using SDCF reduce latency and exhibit better performance in term of cost, energy and mobility pattern for computing resources.

Potluri et al. (2020) aims to improves data analytics and processing by bringing automation controller of IOT healthcare devices at fog network. This can be implemented by incorporating efficient task scheduling model and resource allocation policies in fog computing such that tasks can be scheduled efficiently and reducing processing time and cost. Current research work by Barik et al. (2021) introduces GeoBD<sup>2</sup> to handles unnecessary storage of big data produced by IoT devices. This is done so by applying geospatial de-duplication scheme on fog assisted cloud computing framework where processing and storage of real time geospatial data are efficiently handled.

Pande et al. (2020) focuses on deploying VM with migration technique in vehicular clouds (VCs) to schedule and load balance the underutilized resources of smart vehicles when they sit idle in parking lot, driveways, roadways and streets. A smart cloud service management (SCSM) system is propose to migrate VCs services from hosted VMs of idle smart vehicle to other potential vehicles. To test the efficiency of SCSM system, round-robin (RR) and deficit weighted RR (DWRR) are applied and SCSM perform more efficiently as compared to its counter-part.

To address average resource utilization in vehicular network, Bhoi et al. (2019) focus on issues related to storage as a service (StaaS) in vehicular network by introducing a novel task scheduling policy for heterogeneous vehicular cloud environment (TSP-HVC). TSP-HVC reduces the makespan and maximizes the average resource utilization. Results show that TSP-HVC performs better than min–min and max–min. On the other hand, a large number of VM migrations can lead to wastage of energy and time, which ultimately degrades the performance of the VMs. Pande et al. (2021) introduces resource utilization-aware VM migration (RU-VMM) algorithm to minimize the consumption of energy by efficiently managing the VM resources. The algorithm uses number of metrics to assess the performance. RU-VMM is found to be more efficient than threshold-based algorithm and cumulative sum (CUSUM).

## CLOUDSIM SCHEDULING POLICY

Cloud computing is a thriving research area as it enhances the quality of service (QoS), better performance and ensure continued services. Cloud computing research requires elaborate and detailed process, and experiments are subjected to repetition a few many times. Not only that, the experiments may depend on some other external sources and may need to scale up to meet the demand of the research. Such a demand may shoot up the budget and is cost-ineffective. The best option for such situation is to opt for a good simulator. Nowadays, there are strong and robust simulators that can simulate a real cloud environment portraying networks, storage, hosts, VMs, task requests, servers, various applications and services. They run in a controlled environment where experiments can be conducted and easily repeated (Buyya et al., 2009). Simulated readings, results and analysis are also generated on the spot. Cloudsim, an event driven simulator (Suryateja, 2016), is one such simulators

where one can channels cloud computing research. The simulated data such as execution time, CPU utilization, response time and other parameters can be extracted facilitating more scope for strong analysis and evaluation (Buyya et al., 2009; Suryateja, 2016).

CloudSim provisions two scheduling policy by default. One is the VM scheduling policy at the host level and the other is cloudlet scheduling policy at the VM level (Calheiros et al., 2009, 2011). The VM and the cloudlet's scheduling policy can be classified in terms of space shared (SS) and time shared (TS) provisioning policy as under:

1. **VM's space shared (SS) scheduling:** It is a VM's allocation policy that allocates one or more processing elements (Pes) to a VM and does not allow sharing of Pes. If the PE is allocated to a VM, it cannot be accessed by other VM until the current VM releases it (Cloudsim.org, 2013; Himthani & Dubey, 2019).
2. **M's time shared (TS) scheduling:** It is a VM's allocation policy that allocates one or more PE to a VM and allows sharing of Pes by multiple VMs (Cloudsim.org, 2013). If a PE can fulfil the requirements of multiple VMs simultaneously then multiple VMs will execute in parallel over the same PE or set of Pes (Himthani & Dubey, 2019).
3. **Cloudlet's space shared (SS) scheduling:** The VM in cloudlet's SS scheduling policy takes into consideration that there will be only one cloudlet per VM (Cloudsim.org, 2013). If a cloudlet is assigned to a VM for execution, other cloudlets will be in a waiting list and no other cloudlet will be executed by that VM until the current cloudlet is not dispatched by the VM. Multiple cloudlets cannot execute in parallel over a VM, even if the VM is capable enough to do so (Himthani & Dubey, 2019).
4. **Cloudlet's Time Shared (TS) Scheduling:** In cloudlet's TS scheduling policy the VM executes the cloudlets in a time-shared manner. In this case, multiple cloudlets can execute in parallel over a single VM. This will be possible, if the VM has computing capabilities large enough that it can execute multiple cloudlets in parallel over it (Cloudsim.org, 2013; Himthani & Dubey, 2019).

For implementing the scheduling policy, VM and cloudlet scheduler operate together in the following combination (Chien et al., 2016; Calheiros et al., 2011):

1. **SS-SS:** That is VM is SS at the host level and cloudlet is SS at the VM level respectively.
2. **SS-TS:** That is VM is SS at the host level and cloudlet is TS at the VM level respectively.
3. **TS-SS:** That is VM is TS at the host level and cloudlet is SS at the VM level respectively.
4. **TS-TS:** That is VM is TS at the host level and cloudlet is TS at the VM level respectively.

### Effect of SS and TS Scheduling Policy

To explain the effect of SS and TS scheduling policy, let us take into consideration a dual core host machine. Each core represents a processing element, therefore a dual core machines has two PE. This host machine hosted two VMs, VM1 and VM2 of two cores each. Each VM hosted four tasks units which demand one core each. As each VM requires two PE and each cloudlet demand one PE, the first virtual VM1 hosted tasks t1 to t4 and VM2 hosted tasks t5 to t8 respectively:

1. **Effect of SS-SS Scheduling Policy:** When the scheduling policy for VM and cloudlet is SS, then assignment of the cores and execution of the cloudlets is carried out in a SS manner. When VM is SS, the space is given and possess by the first VM and the VM2 will get its turn only if VM1 completes its execution. In this example, VM1 got activated first and hence the core is assigned to it first. VM2 will get activated only after VM1 finishes the execution of the cloudlets. Since the cloudlet is also provisioned in SS policy, at a time, two cloudlets can run simultaneously in the VM as each cloudlet requires only one core, the remaining cloudlet wait in execution queue

for their turn (Calheiros et al., 2011). Figure 1 illustrates the effects of VMs and cloudlets SS Scheduling Policy.

2. **Effect of TS-SS Scheduling Policy:** In this case, the assignment of the cores to the VM happen on TS and execution of the cloudlets on SS. That is each VM's processing core receives a certain amount of time slice and during that time frame, the cloudlet inside each VM is distributed in a space-shared manner. The VMs are switched between each other as per the time slice assigned. However, next task will be processed only if the previous task is completed. As the cloudlet scheduler is SS, at a time only one task is actively using the processing core. Since, the cores are shared, the amount of processing power available to a VM is determined by calculating VMs that are active on a host (Calheiros et al., 2011) as can be seen in Figure 2.
3. **Effect of SS-TS Scheduling Policy:** When the VM scheduling policy is SS and cloudlet scheduling policy is TS, then space priority is given to current VM and the next VM will get activated only after current VM finishes the execution of the task unit. However, during a VM lifetime, all the tasks assigned to it are dynamically context switched during their life cycle (Calheiros et al., 2011) as can be seen in Figure 3.
4. **Effect of TS-TS Scheduling Policy:** When the VM and cloudlet scheduling policy is TS, a time slice is assigned to the VMs and the processing power is concurrently shared by the VMs. In turn, the cloudlet in the VM distributed this VM's allotted time among themselves and execute in parallel over a single VM accordingly as can be seen in Figure 4. In this case, there are no queuing delays associated with task units (Calheiros et al., 2011).

Figure 1. Effect of VM's and Cloudlet's SS Scheduling Policy

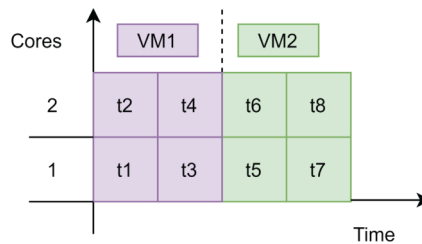


Figure 2. Effect of VM's TS and cloudlet's SS Provisioning Policy

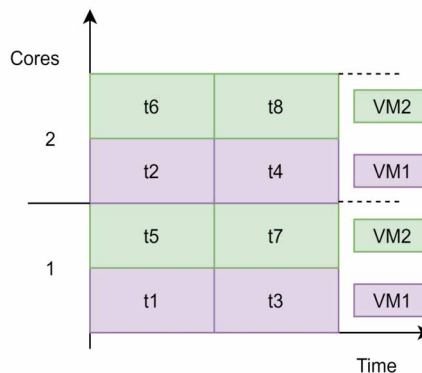


Figure 3. Effect of VM's SS and cloudlet's TS Provisioning Policy

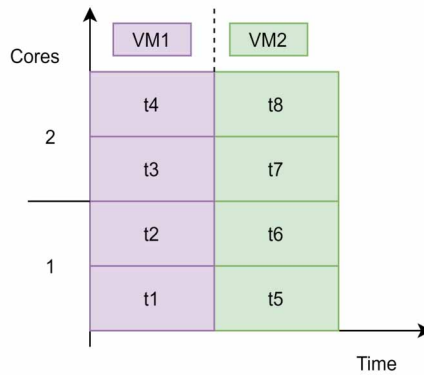
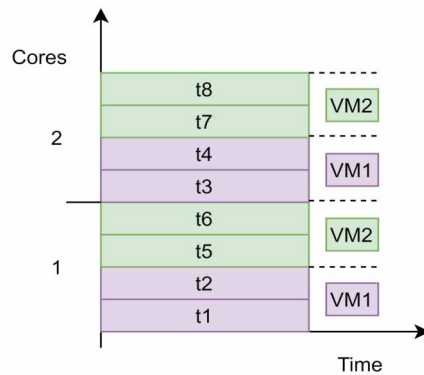


Figure 4. Effect of VM's and cloudlet's TS Scheduling Policy



## OBJECTIVES AND METHODOLOGY OF THE PROPOSED WORK

The main aim of this work is to assess and compare the performance of the response time (RT) and VM's CPU utilization in a cloud computing environment using CloudSim simulator.

### Objectives

The main objectives of this work are:

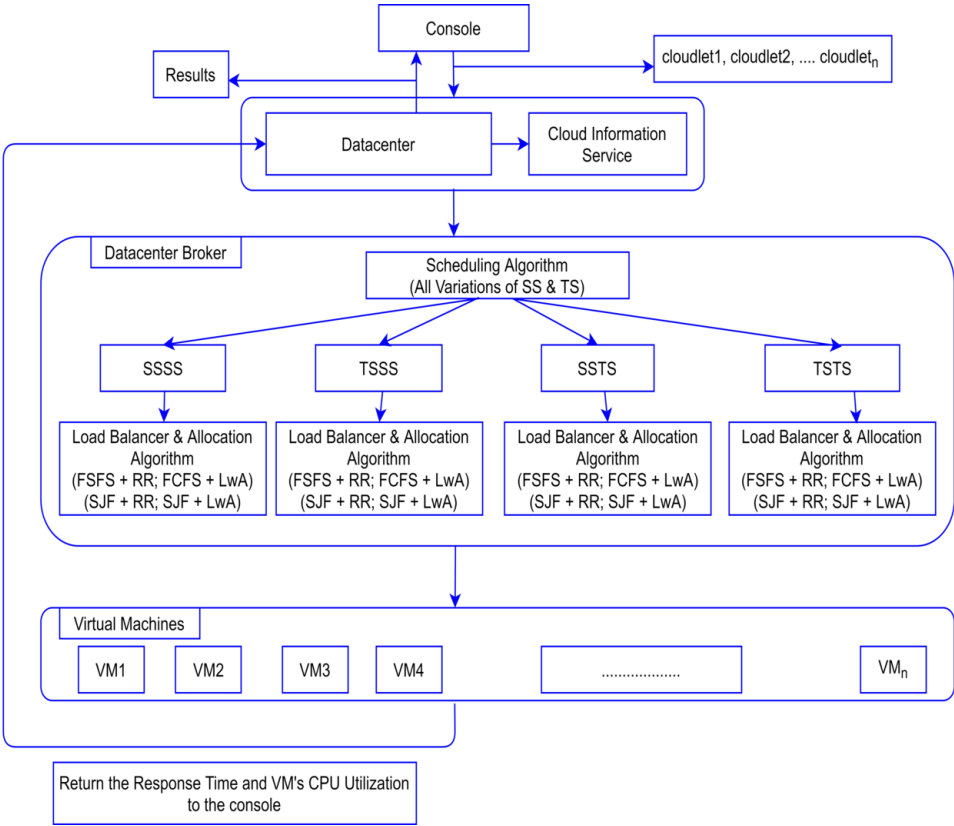
1. To distribute the workload which comprises of a set of 100 and 500 heterogeneous cloudlets from the datacenter broker to VM using a hybrid provisioning, scheduling and allocation technique as can be seen in Table 1 and Figure 5.
2. To assessed and analyzed the performance of a cloud computing system with respect to response time and VM's CPU utilization by each cloudlet in all these hybrid environmental setup as seen in Table 1.

Cloudlet, in this study, represents a task unit or application service (Suryateja, 2016; Calheiros et al., 2009, 2011).

Table 1. Experimental conditions to assess the response time and VM's CPU utilization

Cases	VM Scheduling Policy	Cloudlet Scheduling Policy	Cloudlet Allocation Technique
Case 1	SS	SS	FCFS & RR; FCFS & LwA
			SJF & RR; SJF & LwA
Case 2	SS	TS	FCFS & RR; FCFS & LwA
			SJF & RR; SJF & LwA
Case 3	TS	SS	FCFS & RR; FCFS & LwA
			SJF & RR; SJF & LwA
Case 4	SS	SS	FCFS & RR; FCFS & LwA
			SJF & RR; SJF & LwA

Figure 5. The allocation of cloudlet from the datacentre to the VM under various scheduling and allocation algorithm



METHODOLOGY AND ALGORITHM OF THE CLOUDLET ALLOCATION POLICY

Round Robin Allocation Policy

In RR allocation technique, all cloudlets are allocated to available VM in a cyclical fashion (Roy et al., 2017) as seen in Table 2. Algorithm 1 illustrates the allocation scheme for 500 cloudlets which is implemented in the submitCloudlets() function of the datacentre broker in the Cloudsim simulator.



**Table 2. Round robin allocation scheme for 500 cloudlets**

Hosts	VM	Cloudlet
H1	VM0	$C = \{0, 10, 20, 30, 40, 50, \dots, N\}$ , such that $N = \{10 - x - 1 \mid x \in \mathbb{N} \text{ and } 1 \leq x \leq 50\}$
	VM5	$C = \{5, 15, 25, 35, 45, 55, \dots, N\}$ , such that $N = \{5(-x - 1) \mid x \in \mathbb{N} \text{ and } 1 \leq x \leq 50\}$
H2	VM1	$C = \{1, 11, 21, 31, 41, 51, \dots, N\}$ , such that $N = \{10 - x - 1 + 1 \mid x \in \mathbb{N} \text{ and } 1 \leq x \leq 50\}$
	VM6	$C = \{6, 16, 26, 36, 46, 56, \dots, N\}$ , such that $N = \{5(-x - 1) + 1 \mid x \in \mathbb{N} \text{ and } 1 \leq x \leq 50\}$
H3	VM2	$C = \{2, 12, 22, 32, 42, \dots, N\}$ , such that $N = \{10 - x - 1 + 2 \mid x \in \mathbb{N} \text{ and } 1 \leq x \leq 50\}$
	VM7	$C = \{7, 17, 27, 37, 47, 57, \dots, N\}$ , such that $N = \{5(-x - 1) + 2 \mid x \in \mathbb{N} \text{ and } 1 \leq x \leq 50\}$
H4	VM3	$C = \{3, 13, 23, 33, 43, 53, \dots, N\}$ , such that $N = \{10 - x - 1 + 3 \mid x \in \mathbb{N} \text{ and } 1 \leq x \leq 50\}$
	VM8	$C = \{8, 18, 28, 38, 48, 58, \dots, N\}$ , such that $N = \{5(-x - 1) + 3 \mid x \in \mathbb{N} \text{ and } 1 \leq x \leq 50\}$
H5	VM4	$C = \{4, 14, 24, 34, 44, 54, \dots, N\}$ , such that $N = \{10 - x - 1 + 4 \mid x \in \mathbb{N} \text{ and } 1 \leq x \leq 50\}$
	VM9	$C = \{9, 19, 29, 39, 49, 59, \dots, N\}$ , such that $N = \{5(-x - 1) + 4 \mid x \in \mathbb{N} \text{ and } 1 \leq x \leq 50\}$

#### Algorithm 1. submitCloudlets() for RR Allocation

```

Input: Cloudlet list
Output: Cloudlet submitted in RR fashion
BEGIN
    INITIALISATION
        SET vmIndex = 0
        SET VM = vm
    END INITIALISATION
    READ CloudletList
    READ VMList
    FOR Cloudlet =1 TO n
        IF Cloudlet != vm THEN
            Generate vmIndex from VMCreatedList
        ELSE
            Get VMId assign to the Cloudlet
        END IF
        SET Cloudlet = VMId
        SUBMIT VMId
        //Set the integrity of the RR Cloudlet Allocation
        //by updating next vmIndex
        vmIndex=(vmIndex+1)%sizeofVMList
        UPDATE CloudletSubmittedList
    END FOR
END

```

#### Length-Wise Allocation Policy

LwA policy, is a novel algorithm, in which cloudlet length are categorized and designed for each VM as can be seen on Table 3. The incoming cloudlets are allocated to the VM base on the cloudlet length slab that has been designed for it. Algorithm 2 illustrates LwA policy and is implemented in the submitCloudlets() function of the datacentre broker in CloudSim simulator.

**Table 3. Cloudlet length slabs assign to VM**

Hosts	VM	Cloudlet Length (10000-12000)
H1	VM0	10000 – 10200
	VM5	11001 - 11200
H2	VM1	10201 – 10400
	VM6	11201 - 11400
H3	VM2	10401 – 10600
	VM7	11401 - 11600
H4	VM3	10601 – 10800
	VM8	11601 - 11800
H5	VM4	10801 – 11000
	VM9	11801 - 12000

**Algorithm 2. submitCloudlets() for Length-wise Allocation Policy**

Input: Cloudlet list, vm list

Output: Cloudlet submitted list in LwA

BEGIN

    INITIALISATION

        SET vmIndex = 0

        SET VM as vm

    END INITIALISATION

    FOR Cloudlet = 1 TO n

        IF cloudletLength >=10000 && cloudletLength <=10200

            vmIndex=0;

        ELSE IF cloudletLength >=10200 && cloudletLength <=10400

            vmIndex=1;

        ELSE IF cloudletLength >=10400 && cloudletLength <=10600

            vmIndex=2;

        ELSE IF cloudletLength >=10600 && cloudletLength <=10800

            vmIndex=3;

        ELSE IF cloudletLength >=10800 && cloudletLength <=11000

            vmIndex=4;

        ELSE IF cloudletLength >=11000 && cloudletLength <=12000

            vmIndex=5;

        ELSE IF cloudletLength >=12000 && cloudletLength <=14000

            vmIndex=6;

        ELSE IF cloudletLength >=14000 && cloudletLength <=16000

            vmIndex=7;

        ELSE IF cloudletLength >=16000 && cloudletLength <=18000

            vmIndex=8;

        ELSE IF cloudletLength >=18000 && cloudletLength <=12000

            vmIndex=9;

        // if user didn't bind this cloudlet and it has not been executed yet

        IF CloudletVMID == -1 THEN

```

        Generate vmIndex from VMCreatedList
    ELSE
        Submit cloudlet to specific VMid
        IF vm ==NULL
            VM was not created, print error message
        END IF
    END IF
END IF
UPDATE CloudletSubmittedList
END FOR
END

```

### VM's CPU Utilization

Load balancing is measured in term of load and load performance. Load is assessed by the CPU queue index and CPU utilization while performance is assessed in term of the average response time of (Chien et al., 2016). In this study, performance of virtual machine is measured by the consumption of VM's CPU by each cloudlets at that current time (Ahmad & Khan, 2019; Cloudsim.org, 2013). It is calculated using Equation 1 and measured in percentage. Algorithm 3 highlights the various steps of VM's CPU utilization which is implemented in the submitCloudlets() function of the datacenter broker:

$$currentCPU = vm \cdot getTotalUtilizationOfCpu(CloudSim \cdot clock()) \quad (1)$$

#### Algorithm 3. Vms\_Cpu\_Utilization()

Input: VM's CPU utilization by each cloudlet

Output: Total VM's CPU usage in percentage at a current time (t)

```

BEGIN
    INITIALISATION
        SET Vm = vm
        SET currentCPU=0
        SET currentCPUusage = 0
    END INITIALISATION
    IF vm != null
        UPDATE vm Processing Time
        currentCPU =
            vm.getTotalUtilizationOfCpu(CloudSim.clock());
        currentCPUusage=currentCPU * 100
        PRINT currentCPUusage
    END IF
END

```

### Response Time

Response time is the summation of the processing time and the cost of the task transmission time, queued through the network nodes (Chien et al., 2016). According to (Chien et al., 2016), the response time is calculated as:

$$ResponseTime = FT - AT + TD \quad (2)$$

where, FT represents the finish time of tasks, AT is the arrival time of the tasks and TD is the transmission delay.

If the scheduling policy is SS-SS or TS-SS, according to (Calheiros et al., 2011), FT is calculated as:

$$FT = est(p) + \frac{rl}{Capacity * cores(p)} \quad (3)$$

where  $est(p)$  is the estimated start time of the cloudlet p;  $rl$  is the total number of instructions of the task p;  $cores(p)$  is the number of core or processing elements required by the cloudlet p and Capacity is the average processing power of a core for a cloudlet p.

If the scheduling policy is SS-SS, then according to (Calheiros et al., 2011), capacity is calculated as:

$$Capacity = \sum_{i=1}^{np} \frac{Cap(i)}{np} \quad (4)$$

where  $Cap(i)$  is the processing power of the core 'i',  $np$  is the actual number of core 'i' that the host is considered.

If the scheduling policy is TS-SS, then according to (Chien et al., 2016), capacity is calculated as:

$$Capacity = \frac{\sum_{i=1}^{np} Cap(i)}{Max(\sum_{k=1}^{\beta} \sum_{j=1}^{\gamma} cores(j), np)} \quad (5)$$

where  $\beta$  is the number of VM in the current host,  $\gamma$  is the number of cloudlets running simultaneously in VM<sub>k</sub>,  $cores(j)$  is the number of core that cloudlet j needs.

If the scheduling policy is SS-TS or TS-TS, according to (Calheiros et al., 2011), FT is calculated as:

$$FT = ct + \frac{rl}{Capacity * cores(p)} \quad (6)$$

where  $ct$  represents the current simulation time.

If the scheduling policy is SS-TS, according to (Calheiros et al., 2011), capacity is calculated as:

$$Capacity = \frac{\sum_{i=1}^{np} Cap(i)}{Max(\sum_{j=1}^{\alpha} cores(j), np)} \quad (7)$$

where  $\alpha$  is the total number of cloudlets in VM that contain the cloudlet;  $cores(j)$  is the number of core that cloudlet j needs.

If the scheduling policy is TS-TS, then according to (Chien et al., 2016), capacity is calculated as:

$$Capacity = \frac{\sum_{i=1}^{np} Cap(i)}{Max(\sum_{j=1}^{\delta} cores(j), np)} \quad (8)$$

where  $\delta$  is the total cloudlet of the considered host.

### Cloud Parameters

The simulation is conducted in CloudSim simulator and algorithms are programmed in Java language using CloudSim library (Buyya et al., 2009; Calheiros et al., 2011; Ahmad & Khan, 2019). The simulations include one datacentre, five hosts, two VMs for each host, a total of ten VMs in all. The dataset for the experiment includes one hundred (100) and five hundred (500) heterogeneous cloudlets. The response time and VM's CPU utilization are evaluated on these heterogeneous cloudlet dataset. The cloud parameters are highlighted in the Table 4.

## SIMULATED EXPERIMENTAL ANALYSIS AND RESULTS

### Experimental Analysis and Graphical Representation of Simulated Obtained Result

See Figures 6-15.

## DISCUSSION

The main advantage of this proposed work is that through the hybridisation of the provisioning, scheduling and allocation algorithm while allocating the cloudlet from the datacenter broker to the VM, we are able to present detailed analysis and findings from different angle that support for efficient cloudlet allocation, which have not seen in existing work. The thorough discussion of the analysis can be seen below.

Base on the analysis, the values generated in various environmental setup did not much difference in term of response time and VM's CPU utilization. This can be seen for both cases of one hundred

Table 4. Cloud parameters

Entity	Cloud Parameter	Value
Datacenter	Number of datacenter	1
Host	Number of host	5
	Number of PE on host	4
	MIPS of PE	10000
	RAM (in MB)	4096
	Storage (in MB)	1000000
	Bandwidth (BW) (in Gbps)	10000
VM	Number of VM	10 (2 per Host)
	Number of PE on VM	1
	MIPS (Million Instruction Per Second) of PE	10000
	RAM (in MB)	1024
	Bandwidth	1024
Cloudlet/Task	Number of cloudlet	100, 500
	Length of cloudlet (in MI)	10000-12000 MI
	No of PE requirement	1

Figure 6. RT and CPU Utilization in all cases of cloudlet's and VM's SS and TS provisioning policy for 100 and 500 heterogeneous number of cloudlets

Workload: 100 & 500 Cloudlets					
Point of Analysis: Response Time and VM's CPU Utilization					
		100		500	
		RT	VM's CPU Utilization	RT	VM's CPU Utilization
SSSS	SJF & RR	6.27	0.9	28.05	0.98
	FCFS & RR	6.3	0.9	29	0.98
	FCFS & LwA	6.63	0.8	29.29	0.97
	SJF & LwA	6.65	0.8	29.43	0.96
TSSS	SJF & RR	6.19	0.9	28.08	0.98
	FCFS & RR	6.29	0.9	29.1	0.98
	FCFS & LwA	6.4	0.88	29.39	0.98
	SJF & LwA	6.53	0.8	29.4	0.96
SSTS	SJF & RR	10.68	4.5	53.26	24.5
	FCFS & RR	10.69	4.5	53.35	24.5
	FCFS & LwA	11.57	4.67	55.83	24.96
	SJF & LwA	11.67	3.91	56.25	24.13
TSTS	SJF & RR	10.58	4.5	53.27	24.5
	FCFS & RR	10.83	4.5	53.35	24.5
	FCFS & LwA	11.7	4.68	55.39	24.69
	SJF & LwA	11.77	3.93	55.43	23.74

Figure 7. The simulated allocation of cloudlet to VM representing in FCFS & RR allocation algorithm under SSSS scheduling policy

```

<terminated> catvm_ssss_fcfs_rr_wc [Java Application] C:\Users\lizia\p2\pool\pl
0.1: Broker: Sending cloudlet 9 with length 11282 to VM #0
- Current VM's CPU Usage (in %): 0.0
0.1: Broker: Sending cloudlet 10 with length 11526 to VM #1
- Current VM's CPU Usage (in %): 1.0
0.1: Broker: Sending cloudlet 11 with length 11141 to VM #2
- Current VM's CPU Usage (in %): 1.0
0.1: Broker: Sending cloudlet 12 with length 10520 to VM #3
- Current VM's CPU Usage (in %): 1.0
0.1: Broker: Sending cloudlet 13 with length 11436 to VM #4
- Current VM's CPU Usage (in %): 1.0
0.1: Broker: Sending cloudlet 14 with length 11540 to VM #5
- Current VM's CPU Usage (in %): 1.0
0.1: Broker: Sending cloudlet 15 with length 11800 to VM #6
- Current VM's CPU Usage (in %): 1.0
0.1: Broker: Sending cloudlet 16 with length 11038 to VM #7
  
```

FCFS Implementation

RR Implementation

Figure 8. The simulated allocation of cloudlet to VM representing in FCFS & LwA allocation algorithm under SSSS scheduling policy

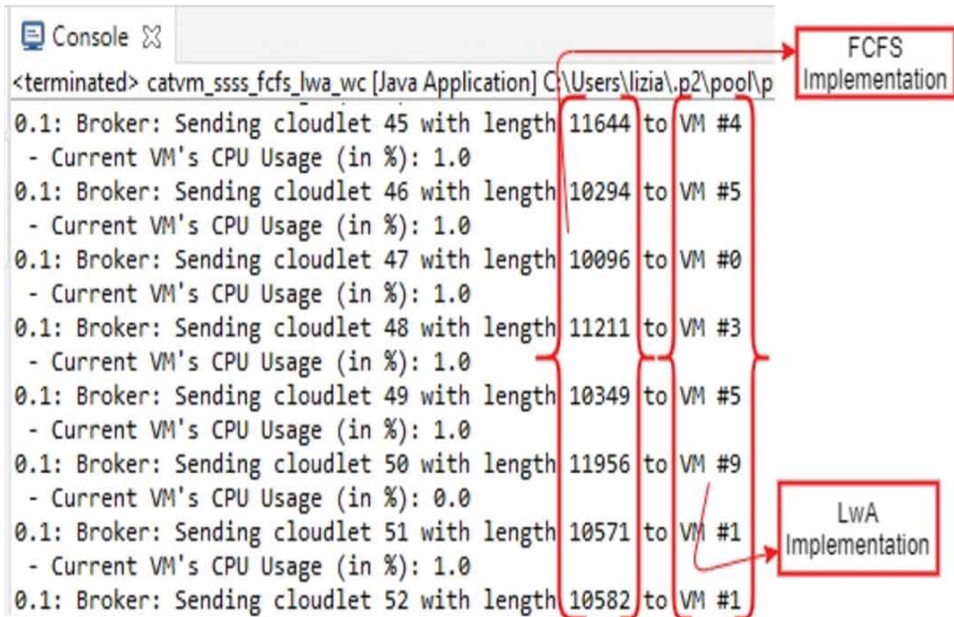


Figure 9. The simulated allocation of cloudlet to VM representing in SJF & RR allocation algorithm under SSSS scheduling policy

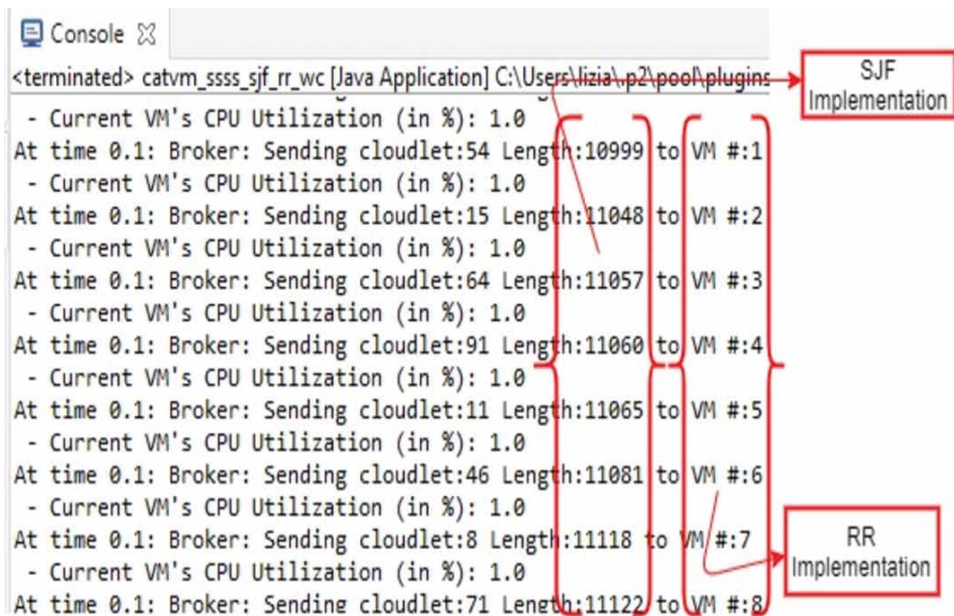




Figure 10. The simulated allocation of cloudlet to VM representing in SJF & LwA allocation algorithm under SSSS scheduling policy

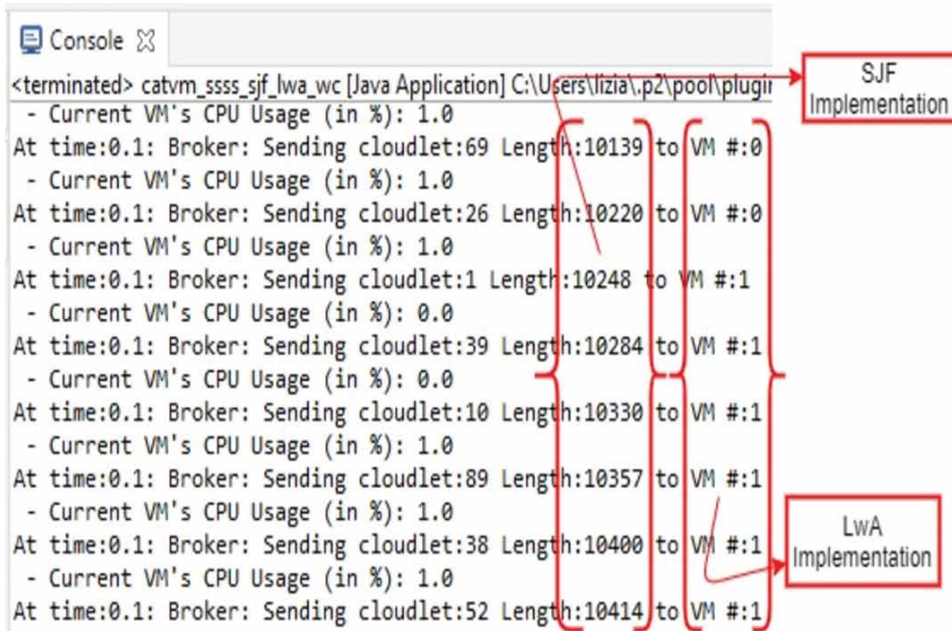
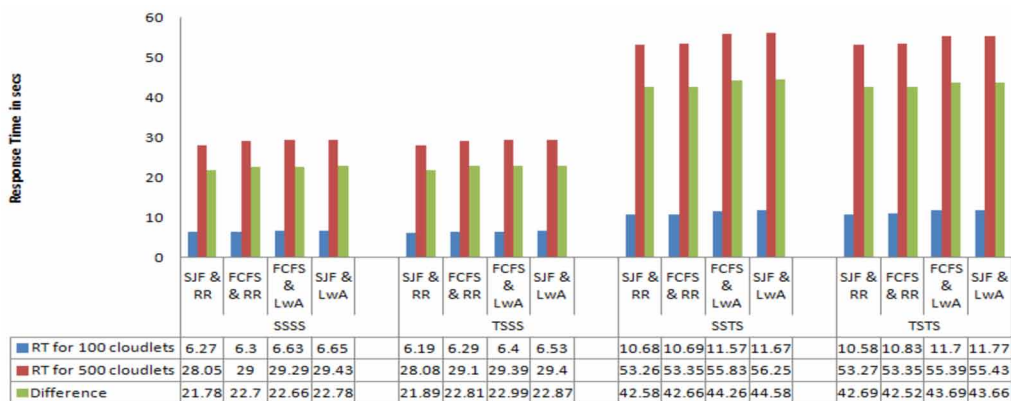


Figure 11. The effect of the response time when 100 and 500 cloudlets are scheduled in all cases of VM's and cloudlet's SS and TS Provisioning Policy and other allocation technique



(100) and five hundred (500) cloudlets. However minimal the case maybe, following are the discussions on the why the response time and VM's CPU utilization behave the way they are:

1. General performance as compare to SS and TS scheduling policy:
  - a. The analysis shows better performance and cost benefit when SS scheduling policy is assigned to the cloudlet at the VM level rather than when TS scheduling policy is assigned to the VM at the host level (Himthani et al., 2019). As seen in Figure 13, the average response time varies from 28.94 secs in SSSS, to 28.99 in TSSS, to 54.36 in TSTS to 54.67 secs in



Figure 12. The effect of the VM's CPU Utilization when 100 and 500 cloudlets are scheduled in all cases of VM's and cloudlet's SS and TS Provisioning Policy and other allocation technique

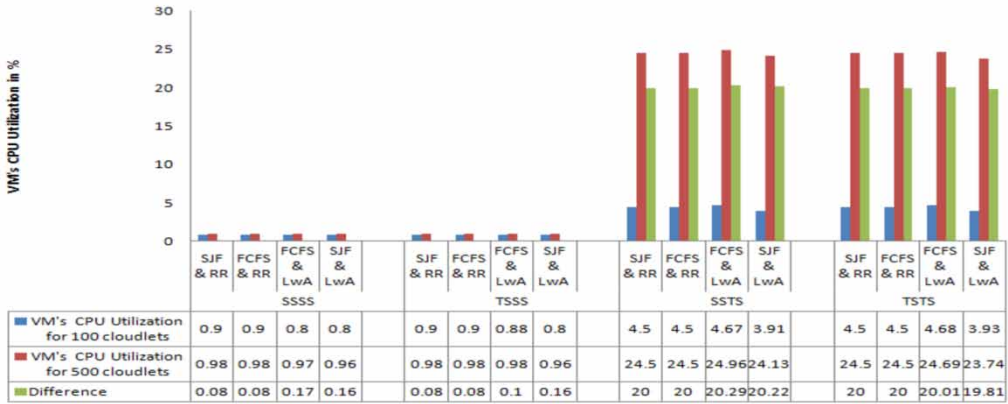


Figure 13. Overall comparison of the average RT and VM's CPU Utilization in all cases of SS and TS provisioning policy

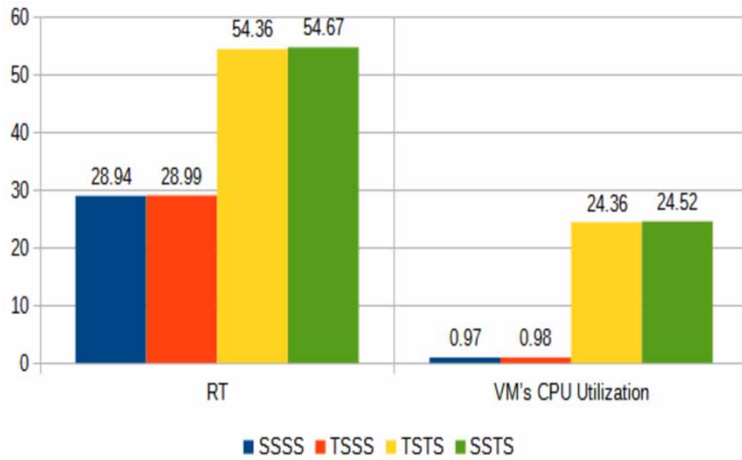


Figure 14. Overall comparison of the average response time when 500 cloudlets are scheduled using SJF & RR, FCFS & RR, FCFS & LwA and SJF & LwA techniques

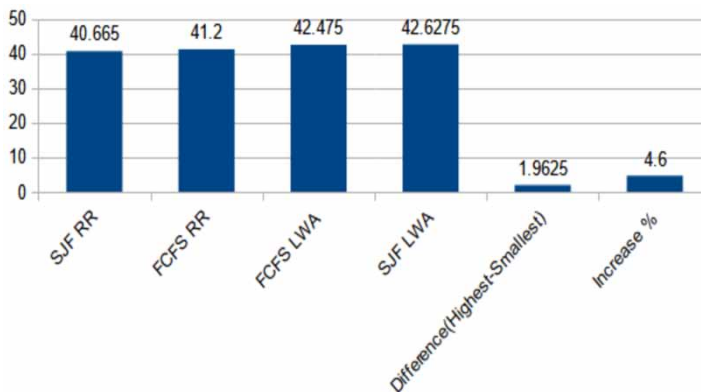
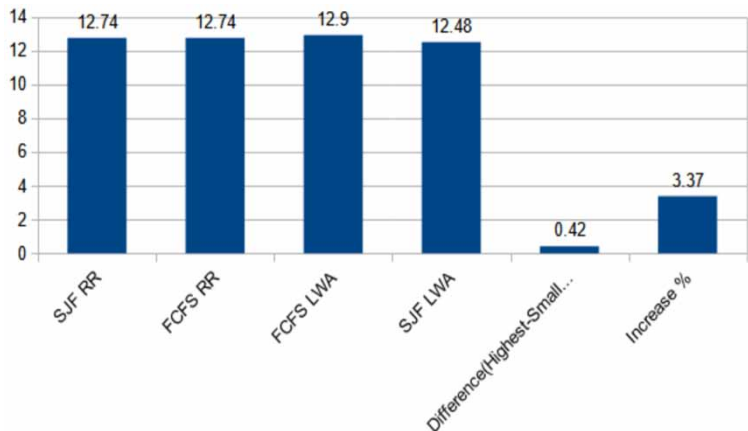


Figure 15. Overall comparison of the average VM's CPU Utilization when 500 cloudlets are scheduled using SJF & RR, FCFS & RR, FCFS & LwA and SJF & LwA techniques



SSTS. Whereas, each cloudlet utilizes the VM's CPU by 0.97% in SSSS, to 0.98% in TSSS, to 24.36% in TSTS and 24.52% in SSTs.

- b. SSSS and TSSS provisioning policy shows least RT with an increase of 46.87% as compare to TSTS and SSTs. This is because when the cloudlets are assigned in a SS manner, the processing elements (PEs) in the VM are not shared and the VM will dispatch that cloudlet only after executing it completely. Because of this behaviour, VM consume less CPU resources of about 96.01% lesser in SSSS and TSSS as compare to TSTS and SSTs as can be seen in Figure 11, 12 and 13.
  - c. SSTs and TSTS provisioning policy shows more RT. This is because when the cloudlets are assigned to VM in a TS manner, PEs in the VM are shared and multiple cloudlets are executing over a single VM, hence the waiting time is more which increases the RT. For the same reason, utilization of CPU resources by the VM is more as compare to SSSS and TSSS.
2. Performance as compare to various allocation policy implemented:
- a. The average RT and VM's CPU Utilization behave differently in different environmental conditions. Arranging in ascending order of performance SJF and RR exhibit good performance at 40.665 secs followed by FCFS & RR and FCFS & LwA at 41.2 and 42.475 secs. The least performance is given by SJF & LwA at 42.6275 secs. Comparing between the least and best, SJF with RR is 4.6% faster as compare to SJF & LwA as seen in Figure 14.
  - b. The cost benefit factor can be quantified based on utilization of resources. Increase CPU utilization increases risk of latency. In this work, VM's CPU Utilization is affected in different environmental setup. Percentage-wise SJF with LwA utilizes least resources at 12.48% followed by SJF & RR and FCFS & RR both at 12.74%. The highest usage can be seen in FCFS with LwA at 12.9%. Comparing between the least and the highest usage, there is an increase of about 3.26% of VM's CPU Utilization between SJF with LwA and FCFS with LwA as seen in Figure 15.

## CONCLUSION

The analysis of this work concludes that the length of the cloudlet and how they are scheduled and allocate to the virtual machine play a major role in determining cost involved and performance. As can be seen, the system exude better performance when SS scheduling policy is assigned to the cloudlet at the VM level rather than when TS scheduling policy is assigned to the VM at the host level. Also,

the SJF along with RR allocation policy shows least RT in all cases of SS and TS provisioning policy for any number of cloudlet. This is because the length of the cloudlet is arranged in ascending order before allocating them to the VM in cyclical RR fashion. However, the performance in FCFS along with RR allocation policy drags down a bit as the length of the cloudlet is not considered even though the cloudlets are allocated to VM in a cyclical RR fashion. In LwA policy, the VM are programmed to take the incoming cloudlet request base on the slab designed for it. Even though the cloudlet is allocated to VM in FCFS or SJF basis, its length and the slab that it belongs to has to be considered before allocating it to VM. Hence more waiting time involves, making this environmental setup slower as compare to the other arrangement. One prominent analysis can be seen in the case of SJF with LwA. SJF with LwA utilizes least VM's CPU, hence very cost effective, but however, shows highest RT which causes more delay. This is the mainly because of the impact of LwA policy. When the cloudlet are assigned from the data center to the VM in SJF with LwA policy, the ready time and processor queuing in the VM decreases, which leads to minimum usage of CPU by each cloudlet in the VM. This analysis contradicts each other in terms of cost and performance. A balance has to be created between utilization and latency which is one of the future scope in this work. Further, we can enhance this work by focussing on contemporary technique, resource migration and green computing which is a demand at present.

## ACKNOWLEDGMENT

The publisher has waived the Open Access Processing fee for this article.

## REFERENCES

- Ahmad, M. O., & Khan, R. Z. (2019). Cloud computing modeling and simulation using CloudSim environment. *International Journal of Recent Technology and Engineering*, 8(2)
- Angela Jennifa Sujana, J., Revathi, T., & Joshua Rajanayagam, S. (2020). Fuzzy-based security-driven optimistic scheduling of scientific workflows in cloud computing. *Journal of the Institution of Electronics and Telecommunication Engineers*, 66(2), 224–241. doi:10.1080/03772063.2018.1486740
- Barik, R. K., Patra, S. S., Patro, R., Mohanty, S. N., & Hamad, A. A. (2021, March). GeoBD2: Geospatial Big Data Deduplication Scheme in Fog Assisted Cloud Computing Environment. In *8th International Conference on Computing for Sustainable Global Development (INDIACom)* (pp. 35–41). IEEE.
- Bhoi, S. K., Panda, S. K., Ray, S. R., Sethy, R. K., Sahoo, V. K., Sahu, B. P., Nayak, S. K., Panigrahi, S., Moharana, R. K., & Khilar, P. M. (2019). TSP-HVC: A novel task scheduling policy for heterogeneous vehicular cloud environment. *International Journal of Information Technology*, 11(4), 853–858. doi:10.1007/s41870-018-0148-6
- Buyya, R., Ranjan, R., & Calheiros, R. N. (2009, June). *Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities*. In *2009 international conference on high performance computing & simulation*. IEEE.
- Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A., & Buyya, R. (2011). CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software, Practice & Experience*, 41(1), 23–50. doi:10.1002/spe.995
- Calheiros, R. N., Ranjan, R., De Rose, C. A., & Buyya, R. (2009). *Cloudsim: A novel framework for modeling and simulation of cloud computing infrastructures and services*. arXiv preprint arXiv:0903.2525.
- Chien, N. K., Son, N. H., & Loc, H. D. (2016, January). Load balancing algorithm based on estimating finish time of services in cloud computing. In *18th International Conference on Advanced Communication Technology (ICACT)* (pp. 228–233). IEEE.
- CloudSim 3.0 API. (2013). *The Cloud Computing and Distributed Systems (CLOUDS) Laboratory, The University of Melbourne*. <http://www.cloudbus.org/cloudsim/doc/api/index.html>
- Deepa, T., & Cheelu, D. (2017, August). A comparative study of static and dynamic load balancing algorithms in cloud computing. In *International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)* (pp. 3375–3378). IEEE. doi:10.1109/ICECDS.2017.8390086
- Himthani, P., & Dubey, G. P. (2019). Performance Analysis of Space Shared Scheduling and Time Shared Scheduling in CloudSim. *International Journal of Recent Development in Engineering and Technology*, 8.
- Hlaing, Y. T. H., & Yee, T. T. (2019, November). Static independent task scheduling on virtualized servers in cloud computing environment. In *International Conference on Advanced Information Technologies (ICAIT)* (pp. 55–59). IEEE. doi:10.1109/AITC.2019.8920865
- Jacob, A., & Raj, C. (2019). Testing Methodologies for Cloud Performance. *International Journal of Innovative Technology and Exploring Engineering*, 8.
- Khaire, G. P. (2017). *Introduction to Cloud Computing*. <https://www.slideshare.net/prakashgkhaire/chapter-2-introduction-to-cloud-computing-79674472>
- Kotas, C., Naughton, T., & Imam, N. (2018, January). A comparison of Amazon Web Services and Microsoft Azure cloud platforms for high performance computing. In *IEEE International Conference on Consumer Electronics (ICCE)* (pp. 1–4). IEEE. doi:10.1109/ICCE.2018.8326349
- Kumar, P., & Silambarasan, K. (2019). Enhancing the performance of healthcare service in IoT and cloud using optimized techniques. *Journal of the Institution of Electronics and Telecommunication Engineers*, 1–10. doi:10.1080/03772063.2019.1654934
- Kumar, R., & Charu, S. (2015). An importance of using virtualization technology in cloud computing. *Global Journal of Computers & Technology*, 1(2).
- Mell, P., & Grance, T. (2011). *The NIST definition of cloud computing*. Academic Press.

- Nithya, S., Sangeetha, M., Prethi, K. A., Sahoo, K. S., Panda, S. K., & Gandomi, A. H. (2020). SDCF: A software-defined cyber foraging framework for cloudlet environment. *IEEE eTransactions on Network and Service Management*, 17(4), 2423–2435. doi:10.1109/TNSM.2020.3015657
- Pande, S. K., Panda, S. K., Das, S., Alazab, M., Sahoo, K. S., Luhach, A. K., & Nayyar, A. (2020). A smart cloud service management algorithm for vehicular clouds. *IEEE Transactions on Intelligent Transportation Systems*.
- Pande, S. K., Panda, S. K., Das, S., Sahoo, K. S., Luhach, A. K., Jhanjhi, N. Z., & Sivanesan, S. et al. (2021). A resource management algorithm for virtual machine migration in vehicular cloud computing. *Computers. Materials & Continua*, 67(2), 2647–2663. doi:10.32604/cmc.2021.015026
- Potluri, S., Sarkar, A., Yasin, E. T., & Mohanty, S. N. (2020). IoT enabled cloud based healthcare system using Fog Computing: A Case Study. *Journal of Critical Reviews*, 7(6).
- Roy, S., Banerjee, S., Chowdhury, K. R., & Biswas, U. (2017). Development and analysis of a three phase cloudlet allocation algorithm. *Journal of King Saud University-Computer and Information Sciences*, 29(4), 473–483. doi:10.1016/j.jksuci.2016.01.003
- Saha, M., Panda, S. K., & Panigrahi, S. (2021). A hybrid multi-criteria decision making algorithm for cloud service selection. *International Journal of Information Technology*, 1-6.
- Shi, Y., Suo, K., Hodge, J., Mohandoss, D. P., & Kemp, S. (2021, January). Towards Optimizing Task Scheduling Process in Cloud Environment. In *IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC)* (pp. 81-87). IEEE. doi:10.1109/CCWC51732.2021.9376146
- Siegel, J., & Perdue, J. (2012, July). Cloud services measures for global use: the service measurement index (SMI). In *2012 Annual SRII global conference* (pp. 411-415). IEEE.
- Suryateja, P. S. (2016). A Comparative Analysis of Cloud Simulators. *International Journal of Modern Education & Computer Science*, 8(4), 64–71. doi:10.5815/ijmecs.2016.04.08
- Vaezi, M., & Zhang, Y. (2017). Cloud mobile networks. Springer.

*Lizia Sahkhar is currently a research scholar in the department of Computer Science & Engineering at National Institute of Technology, Meghalaya (India), where she is working towards her PhD degree. She is also serving as an Assistant Professor at Lady Keane College, Shillong. She received her Masters degree in Information Technology from Sikkim Manipal University, Sikkim, in 2009. Her research interest includes Cloud Computing, Edge Computing and Internet of Things.*

*Bunil Kumar Balabantaray has received his B. Tech. in Information Technology and M.Tech. in Computer Science and Engineering from BPUT, India in the year of 2005 and 2010 respectively. He has completed his PhD from National Institute of Technology Rourkela, India in the year of 2017. Currently he is working as an Assistant Professor in the department of Computer Science and Engineering, National Institute of Technology, Meghalaya. He has 15 years of teaching and research experience. He is serving as reviewer of many journals of national and international repute. His area of research includes IoT, Cloud Computing, Computer Vision, Robotic Vision and Biomedical Image Processing.*

*Satyendra Singh Yadav received his Bachelor of Engineering in Electronics and Communication from Rajiv Gandhi Proudyogiki Vishwavidyalaya (RGPV) a State University of Madhya Pradesh (India), in 2012. In 2018 he received his PhD from National Institute of Technology, Rourkela (India). He was with Instituto de Engenharia e Sistemas Computadores Investigação e Desenvolvimento (INESC-ID), Instituto Superior Técnico Lisbon, Portugal under India-EU NAMASTE mobility project during 2015 to 2016. He is currently working as an Assistant Professor in the department of ECE at National Institute of Technology Meghalaya (India). Prior to joining NIT Meghalaya in Oct. 2019, he has worked as a full-time faculty member at IIITDM Kurnool and IIIT Vadodara. He is actively serving as a reviewer for many IEEE, Springer journals and conferences. His research interests include wireless communication, Resource allocation, Parallel computing, Machine learning, as well as GPU acceleration for 5G and beyond wireless systems. Since 2014, he has been a member of IEEE.*