International Journal of Hybrid Intelligent Systems -1 (2017) 1–10
DOI 10.3233/HIS-170239
IOS Press

An approach towards development of an intelligent cloudlet scheduling mechanism for cloud QoS improvement

Sourav Banerjee[a,*], Akash Chowdhury[b], Swastik Mukherjee[b] and Utpal Biswas[c]
[a]*Department of Computer Science and Engineering, Kalyani Government Engineering College, Kalyani, Nadia*
[b]*Institute of Science and Technology, West Bengal, India*
[c]*University of Kalyani, Kalyani, Nadia*

Abstract. Cloud computing is the ubiquitous on demand service that has brought remarkable revolution in the commercialization of High Performance Computing (HPC) [1]. Quality of Service (QoS) is the vital factor that always seeks high attention. Efficient Resource allocation and management techniques along with advance load balancing approaches make a bigger difference in terms of total system throughput. Several frameworks and algorithmic approaches are proposed in these areas to improve the throughput. Cloudlets are the tasks formed as the requirements of cloud users and submitted to the Local Queues (LQ) of Virtual Machines (VM) by the Datacenter Broker (DCB) to be processed. In this paper the main focus is given to this cloudlet scheduling policy which is nothing but an enhancement of the existing Improved Round Robin Cloudlet Scheduling Algorithm (IRRCSA) and Round Robin Algorithm (RRA). CloudSim 3.0.3 is used to implant the modelling and several parameters like Context Switching (CS), waiting Time (WT), Turnaround Time (TAT) are taken into account to highlight the QoS improvement in comparison with the IRRCSA and RRA approaches.

Keywords: Cloudlet, Local Queue (LQ), Virtual Machine (VM), Quality of Services (QoS), Datacenter Broker (DCB), CloudSim 3.0.3, Improved Round Robin Cloudlet Scheduling Algorithm (IRRCSA), Round Robin Algorithm (RRA)

1. Introduction

Cloud computing is one of the powerful technologies that are growing faster for its on demand services without being worried about the maintenance cost from the perspective of cloud users. It bridges between the requirements of cloud users and services by the cloud provider. This on demand ubiquitous service thus dragging the attention of many researchers to work and improve its each and every area to shape it as more advanced technologies. SLA (Service Level Agreement) [2] is the contract through which the cloud provider and cloud user decides about the services and cost to be taken care of. Cloud users need not to worry about the resource management and maintenance having the flexibility to access in the cloud anytime from anywhere. The services are needed to be provided continuously to meet the needs of cloud users. Thus virtualization of resources takes many technologies together and combines it for better QoS [3]. Several parameters affect the QoS leading towards the need of constant improvements. Several frameworks and algorithmic approaches have been proposed to modify and improve the QoS. From the architectural development to the infrastructural advancements, all revolves around the QoS parameters. Efficient resource allocation and management techniques imply the reliability while advancements in the cloudlet scheduling policies implant the flexibility towards the increased system throughput. Our proposed work brings remarkable advancements over IRRCSA [4] which is highlighted in Section 5.

[*]Corresponding author: Sourav Banerjee, Department of Computer Science and Engineering, Kalyani Government Engineering College, Kalyani, Nadia. E-mail: mr.sourav.banerjee@ieee.org.

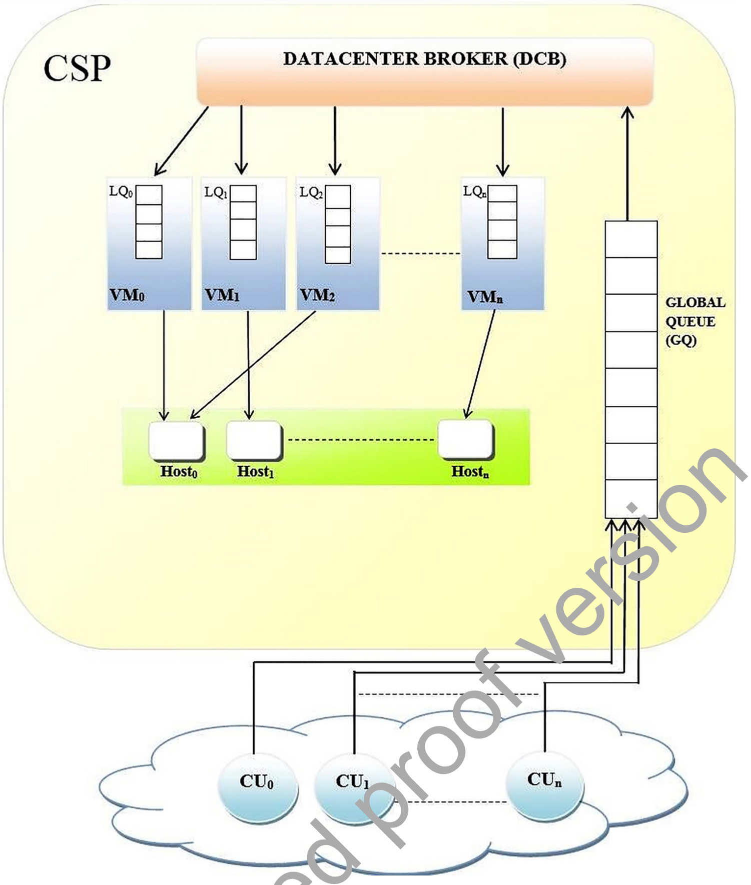

Fig. 1. CloudSim working principle.

1.1. Our contribution

The proposed work modifies the existing cloudlet scheduling algorithm concentrating on the time consuming factor which affects the QoS of the cloud computing. The proposed algorithm in this paper prioritizes shorter cloudlets to be processed first but with a dynamic time quantum along with the Remaining Execution Time (RET) which is elaborated in the Section 3. This hybrid approach brings noteworthy improvements as it reduces the CS, WT and TAT remarkably which further implies the greater system efficiency.

1.2. Organization

The rest of the paper is organized as follow: Section 2 depicts similar works done on this area. Section 3 highlights the evaluation of proposed work elaborating the relevant formulations, algorithm and flowchart. Experimental results are tabulated in the Section 4. Comparison and Analysis section is elaborated in Section 5 which comprises of two sub sections-Tabular analysis and Graphical Analysis. Future research direction and conclusion are discussed in Section 6 and 7 respectively.

2. Related works

The research domain of Cloud Computing [5–7] has involved numerous researchers and scientists all over the world for its different elementary and parametric developments. Many improved cloudlet assignment policies are developed by several researchers for better resource allocation and cloudlet scheduling. The newly created algorithms and policies are mainly simulated in CloudSim [8] framework. GridSim Toolkit [9] provides a system to design and evaluate new scheduling algorithms. It helps in modeling and simulating various Grid resources [10,11], resource broker, schedulers and applications that forms various entities of parallel and distributed computing.

SimGrid [12], GridSim [13], GangSim [14], MicroGrid [15] and EDGSim [16] are some of the well-known grid simulators. Grid simulators though can be used to evaluate the execution costs of distributed applications on cloud infrastructure and address problems related to performance evaluation of heterogeneous grid systems in controlled manner; it does not provide support for seamless simulation, modeling and experimentation systems on cloud computing infrastructure. It also does not support modeling of virtualization enabled resources and application management required on demand [17].

2.1. CloudSim framework

CloudSim is an open source framework that provides seamless modeling, experimentations and simulations possible on cloud infrastructure [18]. CloudSim 3.0.3 is developed on the famous CloudSim [19] framework that is further built on GridSim toolkit [20]. CloudSim framework is a self-contained platform that provides a system to model elements of cloud infrastructure like VMs, datacenters, datacenter broker and policies for improved scheduling and allocation of VMs and cloudlets in large scale cloud platforms. Cloudsim 3.0.3 comprises various abstract and concrete classes for implementation of VM schedulers, Datacenters, Cloudlet scheduler, Datacenter Broker and VM allocation policies. CloudAnalyst is a tool based on cloudsim framework for modelling and analyzing issues related to cloud computing environments [21,22]. Figure 1 depicts the CloudSim working principle.

CloudSim modules that are used in our research are listed below [1].

- VM Scheduler – It is an abstract class that keeps in mind the stated scheduling policy (time shared, space shared) and is implemented by any host component to schedule VMs to Processing Elements (PE) [23].
- Cloudlet Scheduler – It is an abstract class that obeys the scheduling policy and schedules suitable cloudlet residing in the LQ of the VM [24].
- VM Allocation Policy – It is an abstract class used to define required VM allocation policy that will be obeyed to allocate VMs to suitable hosts in datacenters [25].
- Datacenter Broker or DCB – It is a class that encapsulates and exhibits broker properties, that is able to takes care of user requirements and handle various tradeoff among service providers and cloud users [26].

2.2. Scheduling algorithms

2.2.1. RRA

In this algorithm, after the cloudlets are allocated in the corresponding LQs of a particular VM, DCB sets a Time Quantum (TQ) for the cloudlets to be executed. Then the first cloudlet in the LQ starts its execution and gets executed for one TQ after which its execution is preempted and the next cloudlet in the LQ gets a chance to get executed for another TQ. This process goes on until all the cloudlets in the LQ have completed their execution.

2.2.2. IRRCSA

In this algorithm a static TQ for all the VMs are assigned by the DCB and cloudlets are executed accordingly. The RET is taken into account for each complete iteration of the cloudlets till the assigned TQ. If the RET of the currently executed cloudlet is less than or equal to the static TQ then that particular cloudlet is not preempted and is allowed to continue its execution for another TQ.

In the proposed work the drawbacks of the existing scheduling algorithms mentioned have been improved in terms of average WT, average TAT and number of CS. It makes a dynamic approach regarding TQ assignment in each VM as discussed in detail in Section 3.

3. Proposed work

In the cloud computing, a global pool of shared memory is used to avail the resources physically. The list of host with varying configuration is registered to Management Information System (MIS) [28] for the Datacenter Broker (DCB) to access them easily. On the other side, the Cloud Service Provider (CSP) [29] receives request from the Cloud Users (CU) on the basis of demands and requirements. The requirements vary in terms of types of requests, length of files, number of processing elements which are nothing but the tasks included to form the cloudlets. In the CloudSim simulation environment, the modeling is done based on the real time scenarios but implanted virtually. The allocation of cloudlets in the Virtual Machines is done by the Data Centre Broker while the total information regarding the actual hardware configuration of VMs is fetched from the MIS for the proper deployment of the VMs in the Datacenters. After the cloudlet allocation phase, several batches of cloudlets get allocated to the respected local queues of deployed VMs.

TQ has a direct impact on the number of CS according to Eq. (1).

$$\mathrm{TQ} \propto (\mathrm{K_{CS}})^{-1} \tag{1}$$

Where, $\mathrm{K_{CS}}$ implies the number of Context Switching which is inversely proportional to TQ.

The Response Time (RT) is another factor which is highly depended on the TQ as of Eq. (2).

$$\mathrm{TQ} \propto \mathrm{RT_{avg}} \tag{2}$$

$\mathrm{K_{CS}}$ also affects in the total System Throughput ($\mathrm{ST_{tot}}$).

$$\mathrm{ST_{tot}} \propto (\mathrm{K_{CS}})^{-1} \tag{3}$$

Smaller TQ increases $\mathrm{K_{CS}}$ but excessive large TQ may turn the time slicing approach into First Come First Serve (FCFS) scheduling policy which can lead to an adverse effect in terms of system efficiency. Thus a proper TQ needs special attention as it is a highly effective parameter to achieve greater system efficiency.

The proposed work focuses on the cloudlet scheduling policy for the cloudlets residing in the LQs of each VM extending the work of existing IRRCSA algorithm. The proposed algorithm inherits the advantageous features of three different scheduling policies – Shortest Remaining Time First (SRTF) [32], Shortest Job First (SJF) [31] and Round Robin (RR) [32] algorithm. All the cloudlets residing in the LQ of a particular VM get sorted first to prioritize the shortest cloudlets to be executed first. Then a Dynamic Time Quantum (DTQ) is assigned that is obtained from Eq. (6). The DTQ is named so because it varies from one VM to another depending on the Expected Execution Time or ET of cloudlets. After every execution of cloudlets for one DTQ, RET for the cloudlets are deduced according to Eq. (7) and compared with the DTQ. Those cloudlets whose RET is found to be less than or equal to the DTQ, are allowed to get executed for another DTQ.

Our proposed approach has brought noticeable improvement over the existing IRRCSA and RRA algorithm. It has reduced the number of Context Switching (CS) along with the Waiting Time (WT) and Turnaround Time (TAT) in comparison with the existing algorithms. These reductions imply greater improvements in QoS of the total system.

The ET of the cloudlets can be calculated by

$$\mathrm{ET_n} = \mathrm{MI_n}/\mathrm{MIPS_i} \tag{4}$$

Where $\mathrm{ET_n}$ implies the ET of the nth cloudlet residing in the LQ of the ith VM, $\mathrm{MI_n}$ implies the Million Instruction contained by the nth cloudlet and $\mathrm{MIPS_i}$ implies Million Instruction Per Second of the ith VM.

The ET of the cloudlets according to Eq. (4) is calculated after the batch of the cloudlets is allocated to the VMs and then the cloudlets are sorted in ascending order following their ETs. Thus shortest cloudlets will come first for execution.

To assign the DTQ the average ET of the cloudlets is calculated following Eq. (5).

$$(\mathrm{ET_{avg}})_\mathrm{i} = \sum\nolimits_0^\mathrm{N} \mathrm{ET_n}/\mathrm{N} \tag{5}$$

Where $(\mathrm{ET_{avg}})_\mathrm{i}$ is the average ET of all the cloudlets residing in the LQ of the ith VM and N implies the total number of cloudlets in the LQ. After getting the average execution time the DTQ for the ith VM is deduced following Eq. (6).

$$\mathrm{DTQ_i} = \lceil (\mathrm{ET_{max}} + \mathrm{ET_{avg}})_\mathrm{i}/2 \rceil \tag{6}$$

Where $\mathrm{DTQ_i}$ is DTQ for ith VM and $\mathrm{ET_{max}}$ is the ET of the cloudlet with highest ET among all the cloudlets residing in LQ of ith VM.

RET of nth cloudlet is checked, after each iteration of the cloudlets till the assigned DTQ following Eq. (7).

$$(\mathrm{RET})_\mathrm{n} = (\mathrm{ET_n}) - \mathrm{DTQ_i} \tag{7}$$

RET is then compared with DTQ according to Eq. (8).

$$(\mathrm{RET})_\mathrm{n} <= \mathrm{DTQ_i} \tag{8}$$

If the condition is true then the execution of the cloudlets is allowed to continue for another DTQ.

In this way our proposed algorithm improves the performance over the existing RRA and IRRCSA algorithm.

3.1. Proposed algorithm

```
Step 1: START
Step 2: VMs are created and stored in vm_list
Step 3: A batch of cloudlets are assigned in the cloudlet_list
Step 4: All the cloudlets are assigned to the VM in FCFS manner
Step 5: for each VM present in the vm_list{
    While (Local Queue! = NULL) {
        for i = 1 to Number of cloudlets allocated {
            ET of cloudlet_i = MI of cloudlet_i/MIPS_vm
        }
    }
}
```

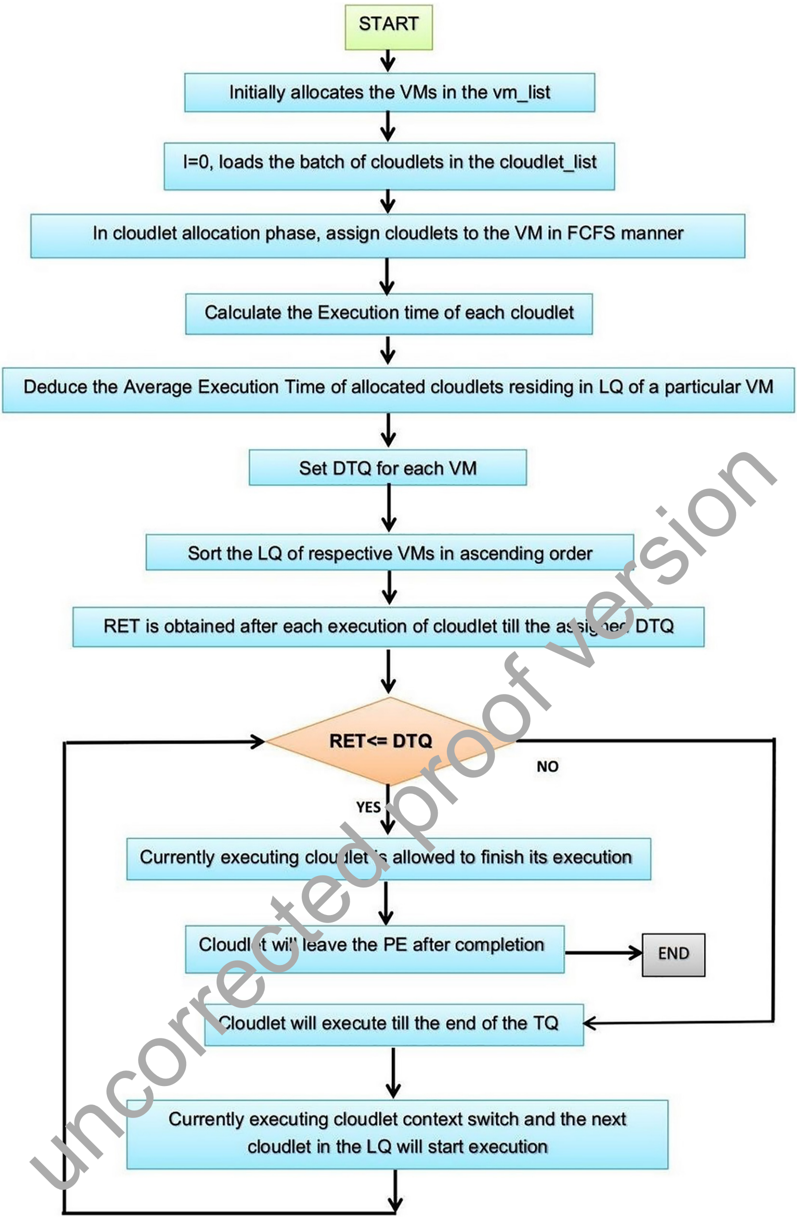
START
Initially allocates the VMs in the vm_list
I=0, loads the batch of cloudlets in the cloudlet_list
In cloudlet allocation phase, assign cloudlets to the VM in FCFS manner
Calculate the Execution time of each cloudlet
Deduce the Average Execution Time of allocated cloudlets residing in LQ of a particular VM
Set DTQ for each VM
Sort the LQ of respective VMs in ascending order
RET is obtained after each execution of cloudlet till the assigned DTQ
RET<= DTQ
NO
YES
Currently executing cloudlet is allowed to finish its execution
Cloudlet will leave the PE after completion
END
Cloudlet will execute till the end of the TQ
Currently executing cloudlet context switch and the next cloudlet in the LQ will start execution

Fig. 2. Flowchart of the proposed work.

```
Step 6: Sort the LQ in ascending order based on the ET of cloudlets
Step 7: Set $ET_{max} = 0$
    for i = 1 to Number of cloudlets allocated {
        If ($ET_{max} < ET_i$) {
            $ET_{max} = ET_i$
        }
    }
Step 8: $ET_{avg} = \sum_0^N ET_n/N$    //where N = Number of cloudlets allocated
    Set DTQ = $\lceil (ET_{max} + ET_{avg})/2 \rceil$
Step 9: While (Local Queue! = NULL){
        for i = 1 to N{
        for j = 1 to DTQ{
            RET = $ET_n$ - DTQ
            if (RET <= DTQ){
                // the current executing cloudlet continue its execution
            }
        Else {
```

Table 1
Properties of cloudlets

Cld_ id	MI
0	9000
1	16000
2	11000
3	6000
4	15000
5	8000
6	12000
7	17000
8	10000
9	7000

Table 2
Properties of VMs

VM_id	MIPS
0	300
1	600
2	400
3	500

Table 3
Cloudlet to VM map table with respective ET

Cld_id	VM_id	ET
0	0	30
1	1	27
2	2	28
3	3	12
4	0	50
5	1	13
6	2	30
7	3	34
8	0	33
9	1	12

Table 4
Allotted cloudlet to VM_0 with their respective ET

Cld_id	MI	ET
0	9000	30
4	15000	50
8	10000	33

```
            // the currently executing cloudlets context switch and the next cloudlet starts execution.
         }
      }
   // the process is continued until all the allotted cloudlets of different VMs present in vm_list did not complete their execution.
   }
}
```

3.2. Flowchart

Flowchart of the proposed work is depicted in Fig. 2.

Table 5
Allotted cloudlet to VM_1 with their respective ET

Cld_id	MI	ET
1	16000	27
5	8000	13
9	7000	12

Table 6
Allotted cloudlet to VM_2 with their respective ET

Cld_id	MI	ET
2	11000	28
6	12000	30

Table 7
Allotted cloudlet to VM_3 with their respective ET

Cld_id	MI	ET
3	6000	12
7	17000	34

Table 8
WT, TAT and Number of CS in VM_0

Cld_id	WT	TAT
0	0	30
8	30	63
4	63	113
Average WT	31	
Average TAT	68.67	
Number of CS	2	

Table 9
WT, TAT and Number of CS in VM_1

Cld_id	WT	TAT
9	0	12
5	12	25
1	25	52
Average WT	12.33	
Average TAT	29.67	
Number of CS	2	

4. Experimental results

We have considered ten cloudlets and four VMs in this example. Table 1 represents cloudlet properties which comprises the cloudlet ids and their respective MIs. Table 2 shows the properties of VMs comprising the VM ids and their respective MIPS. Table 3 shows the cloudlets being mapped to particular VMs according to First Come First Serve or FCFS manner and thus consists of Cloudlet ids or Cld_ids with their respective VM ids and ETs. In Tables 4–7 cloudlets allocated to each VM (0, 1, 2, 3) are shown respectively. In VM_0 according to the allocated cloudlets and Eq. (6) we have obtained the DTQ to be 44. Table 8 depicts the Waiting Time, Turnaround Time and Number of Con-

Table 10
WT, TAT and Number of CS in VM_2

Cld_id	WT	TAT
2	0	28
6	28	58
Average WT	14	
Average TAT	43	
Number of CS	1	

Table 11
WT, TAT and Number of CS in VM_3

Cld_id	WT	TAT
3	0	12
7	12	46
Average WT	6	
Average TAT	29	
Number of CS	1	

text Switching obtained in VM_0. In VM_1 according to the allocated cloudlets and Eq. (6) we have obtained the DTQ to be 22. Table 9 depicts the Waiting Time, Turnaround Time and Number of Context Switching obtained in VM_1. In VM_2 according to the allocated cloudlets and Eq. (6) we have obtained the DTQ to be 30. Table 10 depicts the Waiting Time, Turnaround Time and Number of Context Switching obtained in VM_2. In VM_3 according to the allocated cloudlets and Eq. (6) we have obtained the DTQ to be 29. Table 11 depicts the Waiting Time, Turnaround Time and Number of Context Switching obtained in VM_3.

5. Comparison and Analysis

In this section the results of RRA, IRRCSA and the PROPOSED algorithms will be compared to each other and analyzed in two ways

5.1) Tabular Analysis
5.2) Graphical Analysis

In Section 5.1, Tables 12, 15, 18 and 21 depict the average WT of VM_0, VM_1, VM_2 and VM_3 respectively along with the Rate of Improvements shown by the proposed algorithm in comparison to RRA and IRRCSA regarding average WT. Tables 13, 16, 19 and 22 depict the average TAT of VM_0, VM_1, VM_2 and VM_3 respectively along with the Rate of Improvements shown by the proposed algorithm in comparison to RRA and IRRCSA regarding average TAT. Tables 14, 17, 20 and 23 depict the number of CS in VM_0, VM_1, VM_2 and VM_3 respectively along with the Rate of Improvements shown by the proposed algorithm in comparison to RRA and IRRCSA regarding number of CS. Table 24 shows a summary of the improvement achieved in the proposed algorithm in comparison to RRA and IRRCSA.

In Section 5.2, Figs 3–5 represent the comparison of WT, TAT and number of CS respectively in all the four VMs in case of RRA, IRRCSA and Proposed algorithm.

5.1. Tabular analysis

Table 12
WT for RRA, IRRCSA and PROPOSED in VM_0

Cld_id	RRA	IRRCSA	PROPOSED
0	50	40	0
4	63	63	63
8	65	55	30
Average WT	59.33	52.67	31
Rate of Improvement (in comparison to RRA)		47.74%	
Rate of Improvement (in comparison to IRRCSA)		41.14%	

Table 13
TAT for RRA, IRRCSA and PROPOSED in VM_0

Cld_id	RRA	IRRCSA	PROPOSED
0	80	70	30
4	113	113	113
8	98	93	63
Average TAT	97	92	68.67
Rate of Improvement (in comparison to RRA)		29.20%	
Rate of Improvement (in comparison to IRRCSA)		25.36%	

Table 14
Number of CS in RRA, IRRCSA and PROPOSED in VM_0

RRA	IRRCSA	PROPOSED
20	17	2
Rate of Improvement (in comparison to RRA)		90%
Rate of Improvement (in comparison to IRRCSA)		88.23%

Table 15
WT for RRA, IRRCSA and PROPOSED in VM_1

Cld_id	RRA	IRRCSA	PROPOSED
1	25	25	25
5	25	15	12
9	28	23	0
Average WT	26	21	12.33
Rate of Improvement (in comparison to RRA)		52.58%	
Rate of Improvement (in comparison to IRRCSA)		41.28%	

Table 16
TAT for RRA, IRRCSA and PROPOSED in VM$_1$

Cld_id	RRA	IRRCSA	PROPOSED
1	52	52	52
5	38	28	25
9	40	35	12
Average TAT	43.33	38.33	29.67
Rate of Improvement (in comparison to RRA)		27.69%	
Rate of Improvement (in comparison to IRRCSA)		22.59%	

Table 17
Number of CS in RRA, IRRCSA and PROPOSED in VM$_1$

RRA	IRRCSA	PROPOSED
9	6	2
Rate of Improvement (in comparison to RRA)		77.77%
Rate of Improvement (in comparison to IRRCSA)		66.67%

Table 18
WT for RRA, IRRCSA and PROPOSED in VM$_2$

Cld_id	RRA	IRRCSA	PROPOSED
2	25	20	0
6	28	28	28
Average WT	26.5	24	14
Rate of Improvement (in comparison to RRA)		47.16%	
Rate of Improvement (in comparison to IRRCSA)		41.66%	

Table 19
TAT for RRA and IRRCSA in VM$_2$

Cld_id	RRA	IRRCSA	PROPOSED
2	53	48	28
6	58	58	58
Average TAT	55.5	53	43
Rate of Improvement (in comparison to RRA)		22.52%	
Rate of Improvement (in comparison to IRRCSA)		18.86%	

Table 20
Number of CS in RRA, IRRCSA and PROPOSED in VM$_2$

RRA	IRRCSA	PROPOSED
11	9	1
Rate of Improvement (in comparison to RRA)		90.90%
Rate of Improvement (in comparison to IRRCSA)		88.88%

Table 21
WT for RRA, IRRCSA and PROPOSED in VM$_3$

Cld_id	RRA	IRRCSA	PROPOSED
3	10	5	0
7	12	12	12
Average WT	11	8.5	6
Rate of Improvement (in comparison to RRA)		45.00%	
Rate of Improvement (in comparison to IRRCSA)		29.41%	

Table 22
TAT for RRA, IRRCSA and PROPOSED in VM$_3$

Cld_id	RRA	IRRCSA	PROPOSED
3	22	17	12
7	46	46	46
Average TAT	34	31.5	29
Rate of Improvement (in comparison to RRA)		14.70%	
Rate of Improvement (in comparison to IRRCSA)		7.93%	

Table 23
Number of CS in RRA, IRRCSA and PROPOSED in VM$_3$

RRA	IRRCSA	PROPOSED
5	3	1
Rate of Improvement (in comparison to RRA)		80%
Rate of Improvement (in comparison to IRRCSA)		66.67%

Table 24
Summery table for Rate of Improvement (%)

Performance measure	VM	Rate of improvement (%) (In Comparison to RRA)	Rate of improvement (%) (In Comparison to IRRCSA)
WT	0	47.74%	41.14%
TAT	0	29.20%	25.36%
CS	0	90%	88.23%
WT	1	52.58%	41.28%
TAT	1	27.69%	22.59%
CS	1	77.77%	66.67%
WT	2	47.16%	41.66%
TAT	2	22.52%	18.86%
CS	2	90.90%	88.88%
WT	3	45%	29.41%
TAT	3	14.70%	7.93%
CS	3	80%	66.67%

5.2. *Graphical analysis*

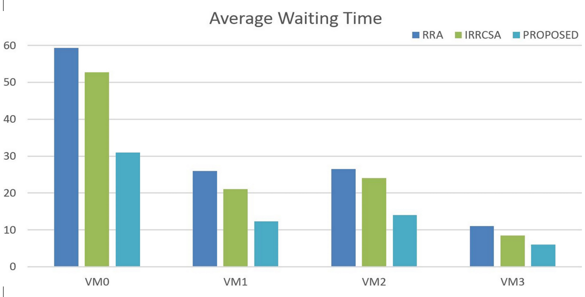

Fig. 3. Average WT for RRA, IRRCSA and PROPOSED in VM_0, VM_1, VM_2 and VM_3.

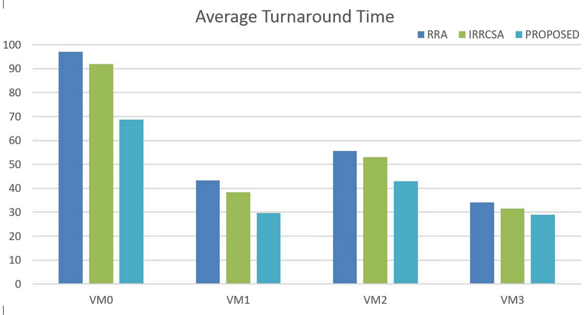

Fig. 4. Average TAT for RRA, IRRCSA and PROPOSED in VM_0, VM_1, VM_2 and VM_3.

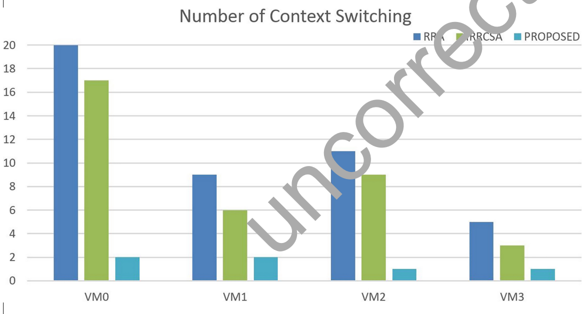

Fig. 5. Number of CS for RRA, IRRCSA and PROPOSED in VM_0, VM_1, VM_2 and VM_3.

6. Future research direction

Cloudlet scheduling policy [30] is the important part leading towards the need for more advance algorithmic approaches focusing on the time shared and space shared module. Besides, formation of newer frameworks which deal with the newer dimensions of this area should always be taken under consideration. Load balancing and optimization of resource allocation techniques are also the fields that are needed to be focused by the researchers. The critical issues like scalability, fault tolerance of the entire system have been considered for future work which will make the approach more robust.

7. Conclusion

The virtualization of the resources and flexible on demand services has leaded cloud computing towards the greatest heights in comparison with other newer technologies. Thus our proposed work has the potential to include the advancements to the cloud computing technology in terms of system efficiency and QoS improvement. We will extend our work further to form more advanced and efficient approaches considering newer dimensions.

References

[1] A.J. Younge, R. Henschel, J.T. Brown, G.V. Laszewski, J. Qiu and G.C. Fox, Analysis of virtualization technologies for high performance computing environments, in: *Cloud Computing (CLOUD), 2011 IEEE International Conference*, 2011, pp. 9–16.

[2] C.J. Ough, in: *Toward Better Understanding of Service Level Agreement (SLA) Expectations for Improved Conformance among Stakeholders in a Distributed, Dynamic, Composite Software Solution, Capstone Report, University Of Oregon*, 2008, pp. 1–80.

[3] R.N. Calheiros, R. Ranjan and R. Buyya, Virtual machine provisioning based on analytical performance and QoS in cloud computing environments, in: *2011 International Conference on Parallel Processing (ICPP)*, 2011, pp. 295–304.

[4] S. Banerjee, M. Adhikari and U. Biswas, Design and analysis of an efficient QoS improvement policy in cloud computing, *Service Oriented Computing and Applications*, Springer **11**(1) (2017), 65–73.

[5] C. Weinhardt, A. Anandasivam, B. Blau, N. Borissov, T. Meinl, W. Michalk and J. Stößer, Cloud computing – a classification, business models, and research directions, *Business & Information Systems Engineering* **1**(5) (2009), 391–399.

[6] F. Liu, J. Tong, J. Mao, R. Bohn, J. Messina, L. Badger and D. Leaf, NIST cloud computing reference architecture, *NIST Special Publication* (2011), 1–28.

[7] W. Voorsluys, J. Broberg and R. Buyya, Introduction to Cloud Computing, *Cloud Computing: Principles and Paradigms* (2011), 1–41.

[8] R.N. Calheiros, R. Ranjan, A. Beloglazov, C.A. De Rose and R. Buyya, CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, *Software: Practice and Experience* **41**(1) (2011), 23–50.

[9] R. Buyya and M. Murshed, Gridsim: A toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing, *Concurrency and Computation: Practice and Experience* **14** (2002), 13–15.

[10] K.M. Sim, Grid resource negotiation: Survey and new directions, *IEEE Transactions on Systems, Man, and Cybernetics* **40**(3) (2010), 245–257.

[11] K.M. Sim, A survey of bargaining models for grid resource allocation, *ACM SIGecom Exchanges* **5**(5) (2006), 22–32.

[12] H. Casanova, Simgrid: A toolkit for the simulation of application scheduling, *Proceedings First IEEE/ACM International Symposium on Cluster Computing and the Grid* (2001), 430–437.

[13] M. Murshed, R. Buyya and D. Abramson, Gridsim: A toolkit for the modeling and simulation of global grids, *Monash University Journal* (2001), 1–5.

[14] C.L. Dumitrescu and I. Foster, GangSim: a simulator for grid scheduling studies, *2005 IEEE International Symposium on Cluster Computing and the Grid* **2** (2005), 1151–1158.

[15] R.H. Lasseter and P. Paigi, Microgrid: A conceptual solution, in: *2004 IEEE 35th Annual Power Electronics Specialists Conference (IEEE Cat. No.04CH37551)*, **6**, 2004, pp. 4285–4290.

[16] C. Dobre, A General Framework for the Modeling and Simulation of Grid and P2P Systems, *Handbook of Research on P2P and Grid Systems for Service-Oriented Computing: Models, Methodologies and Applications*, IGI Global (2010), 657–685.

[17] R. Malhotra and P. Jain, Study and comparison of various cloud simulators available in the cloud computing, *International Journal of Advanced Research in Computer Science and Software Engineering* **3**(9) (2013), 347–350.

[18] R. Buyya, R. Ranjan and R.N. Calheiros, Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities, in: *2009 International Conference on High Performance Computing & Simulation*, 2009, pp. 1–11.

[19] R.N. Calheiros, R. Ranjan, C.A. De Rose and R. Buyya, Cloudsim: A novel framework for modeling and simulation of cloud computing infrastructures and services, arXiv e-print arXiv:0903.2525, 2009, 1–9.

[20] R. Buyya and M. Murshed, GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing, arXiv:cs/0203019, Concurrency and Computation: Practice and Experience, Wiley, 2002, 1–37.

[21] B. Wickremasinghe and R. Buyya, Cloudanalyst: A cloudsim-based tool for modelling and analysis of large scale cloud computing environments, *MEDC Project Report* (2009), 433–659.

[22] B. Wickremasinghe, R.N. Calheiros and R. Buyya, Cloudanalyst: A cloudsim-based visual modeller for analysing cloud computing environments and applications, in: *24th IEEE International Conference on Advanced Information Networking and Applications*, 2010, pp. 446–452.

[23] T. Chatterjee, V.K. Ojha, M. Adhikari, S. Banerjee, U. Biswas and V. Snášel, Design and Implementation of an Improved Datacenter Broker Policy to Improve the QoS of a Cloud, in: *Proceedings of the Fifth International Conference on Innovations in Bio-Inspired Computing and Applications IBICA 2014, AISC 303, Springer*, 2014, pp. 281–290.

[24] S. Banerjee, M. Adhikari, S. Kar and U. Biswas, Development and analysis of a new cloudlet allocation strategy for QoS improvement in cloud, *Arabian Journal for Science and Engineering* **40**(5) (2015), 1409–1425.

[25] M. Adhikari, S. Banerjee and U. Biswas, Smart task assignment model for cloud service provider, in: Special Issue of International Journal of Computer Applications (0975–8887) on Advanced Computing and Communication Technologies for HPC Applications – ACCTHPCA, 2012, 43–46.

[26] S. Banerjee, M. Adhikari and U. Biswas, Development of a smart cloudlet allocation model for a cloud service provider, in: *2nd International Conference on Business and Information Management (ICBIM)*, 2014, pp. 114–119.

[27] A. Ahmed and A.S. Sabyasachi, Cloud computing simulators: A detailed survey and future direction, in: *IEEE International Advance Computing Conference (IACC)*, 2014, pp. 866–872.

[28] L. Wang, Y. Lan and Q. Xia, Using cloudsim to model and simulate cloud computing environment, in: *2013 Ninth International Conference on Computational Intelligence and Security*, 2013, pp. 323–328.

[29] Y. Wei and M.B. Blake, Service-oriented computing and cloud computing: Challenges and opportunities, *IEEE Internet Computing* **14**(6) (2010), 72–75.

[30] S. Sindhu and S. Mukherjee, Efficient task scheduling algorithms for cloud computing environment, in: A. Mantri, eds, *High Performance Architecture and Grid Computing*, CCIS 169, Springer-Verlag Berlin Heidelberg, 2011, pp. 79–83.

[31] J. Ru and J. Keung, An empirical investigation on the simulation of priority and shortest-job-first scheduling for cloud-based software systems, in: *2013 22nd Australian Software Engineering Conference*, 2013, pp. 78–87.

[32] M. Thombare, R. Sukhwani, P. Shah, S. Chaudhari and P. Raundale, Efficient implementation of Multilevel Feedback Queue Scheduling, in: *2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, 2016, pp. 1950–1954.