Turkish Journal of Computer and Mathematics Education Vol.12 No.10 (2021), 5629-5638

Research Article

Enhanced Time Quantum based Round Robin Algorithm for Cloudlet Scheduling in Cloud Environment

D.Gritto [1], Dr.P.Muthulakshmi [2]
Research Scholar [1], Associate Professor [2]
1, 2 - Department of Computer Science, CSH, SRM Institute of Science and Technology, Chennai, India.
grittodg@gmail.com [1], muthulap@srmist.edu.in [2].

Article History: Received: 10 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 28 April 2021

Abstract – The contemporary organization integrates most of the required resources into their business environment by adopting or leasing the cloud resources that are offered by many of the remote cloud service providers. The cloud computing renders hassle free installation, deployment and maintenance of the resources. The cloud resources and services are observed to be pervasive in nature and therefore efficient technologies are to be adopted in order ensure the effective resource utilization and optimal service delivery to the cloud users. Generally virtualization, optimal scheduling techniques like cloudlet scheduling, virtual machine scheduling and load balancing strategies maximize the resource utilization and service delivery. This paper outlines the foundation concepts of cloud computing and proposes an algorithm named Enhanced Time Quantum based Round Robin Algorithm (ETQRRA). The ETQRRA is proposed to reduce the response time, waiting time, turnaround time and the number of context switching. The result of ETQRRA algorithm is compared with various versions of the existing Round Robin algorithms like Classic Round Robin Algorithm (CRR), Improved Shortest Remaining Burst Round Robin (ISRBRR), Half Life Variable Quantum Time Round Robin (HLVQTRR), Improved Round Robin Varying Quantum (IRRVQ) and Dynamic Time Quantum Round Robin (DTQRR) algorithms to prove the efficiency of ETQRRA. The comparative study shows the performance of ETQRRA outdo the other algorithms taken for the comparison.

Index Terms – Virtualization, Scheduling, Scheduling Algorithms, Cloudlet Scheduling, Virtual Machine Scheduling, Load Balancing

I. Introduction

Cloud computing is a form of distributed computing that offers large scale hosting and sharing of resources on a server that is located in a server farm, globally from any geographical area. The user can build extremely huge virtual infrastructure by assessing the resources like CPU, memory, bandwidth, software, hardware and other utilities from the remote server or data centre of a Cloud Service Provider (CSP). These resources are accessed via the network through an on-demand basis by paying the subscription charges. The prominent service models of cloud are Software as Service (SaaS), Platform as Service (PaaS) and Infrastructure as Service (IaaS) [1]. Major deployment models are Private cloud, Public cloud, Community cloud and Hybrid cloud.

In recent times, the cloud users are increasing exponentially and therefore massive resources with huge capacity is needed to be deployed in every data centre. These resources are delivered to the cloud client as virtualized resources. Virtualization is a process of creating virtual instances of hardware, software, storage and network devices [2]. Virtualization divides the physical servers into a set of executable virtual environments called the virtual machines. The Virtual Machines (VMs) are created by the virtual machine manager called the hypervisor. Hypervisor creates the VMs as self provisioned system and allows the VMs to run in isolation. The hypervisor places itself above the hardware or operating system and manages the virtual machines by sharing the physical resources to each virtual machine and segregates one machine from the other. Virtualization leads to cost reduction, ease of cloudlet execution, reduced data migration, increased fault tolerance and maximized resource utilization.

The huge cloudlets or client request may persuade the cloud environment at any instant of time. This leaves a high risk for the CSPs to manage the cloudlets and the resources. Both the cloudlets and the virtual instance of resources can be managed effectively through professionally devised scheduling algorithms. Scheduling is a process of mapping a group of cloudlets to a group of VMs or VMs to set of physical host (processor core) in such a way that it optimizes the performance and the resource utilization [3]. The cloud scheduler can be static or dynamic in nature. Static scheduler has predefined sequence for the mapping of the cloudlets and the VMs and this type of scheduling is called as compile time or offline scheduling. The dynamic scheduler schedules the cloudlets to the VMs during run time or it can be done online (on the go fashion). The traditional scheduling algorithms that are extensively used for the scheduling includes, First Come First Serve (FCFS), Shortest Job First (SJF), Shortest Remaining Time First (SRTF), Round Robin (RR), and Earliest Deadline First (EDF). In recent times, the

implementation of heuristic and meta heuristic based optimization algorithms like Genetic Algorithm, Simulated Annealing, Swarm Intelligence algorithms, Fuzzy and Neural Network based algorithms are used extensively for scheduling in the cloud computing. Load balancing technology manages the resources by effectively dispersing the load across a group of servers to maintain the equilibrium of the load among the servers. The load balancer distributes the load across servers/virtual machines and identifies the servers/virtual machines that are overloaded or under loaded. The resource requirement of the user also varies over time therefore it is required to have nominal number of robust resource handling technologies for dynamic provisioning and de-provisioning of the resources.

Round Robin algorithm is a pre-emptive scheduling algorithm that is based on time slice and rounds of execution. Each cloudlet in the VM ready queue is given a fair chance for execution in each round and it is found that this policy improves the response time of cloudlets. Here, a set of cloudlets are allocated to a set of selected VMs in a cyclic order. The challenge associated with the RR algorithm is the estimation of right time quantum for all iterations. The time quantum decides the performance of the scheduling algorithm. Also, RR algorithm does not consider the current load limit of the VM prior to the allocation of cloudlets. It is found that the sequel version of the RR algorithm called the Weighted Round Robin (WRR) overcomes this challenge. WRR estimates the capacity of each VM and allocates the cloudlets in multiples as per the capacity of the specific VM. This improves the load distribution compared to that of the traditional RR algorithm.

This paper proposes an effective time quantum estimation method to optimize the cloudlet scheduling. The Enhanced Time Quantum based Round Robin Algorithm (ETQRRA) for cloudlet scheduling is evaluated and it has produced an improved result. The paper is further organized as, section II is presented with the literature review some related works. Section III enumerates the scheduling criteria. Section IV and V comprises of the review of the algorithms taken for comparison and overview regarding the simulation environment considered respectively. Section VI contains the description of the proposed algorithm. Section VII includes the Results and Discussion of the work and Section VIII corresponds to the conclusion of the study.

II. Literature Review

1: Dynamic Multilevel Hybrid Scheduling Algorithm for Grid Computing

Syed Nasir Mehmood Et al. proposed two grid scheduling algorithms that minimize the waiting time, turnaround time and the response time [4]. The substrate of the two proposed algorithms are Dynamic Multilevel Hybrid Scheduling Algorithm using Median (MHM) and Dynamic Multilevel Hybrid Scheduling Algorithm using Square root (MHR) is the Multilevel Hybrid Scheduling Algorithm (MH). The MH algorithm adopts the master slave architecture with master processor controls the process of scheduling the gridlets to the slave processing elements (PEs) in a Round Robin fashion. The MHR algorithm ascends the tasks based on their execution time. The MHR makes use of two variables T_{LARGE} (largest task execution time) and T_{EXE} (total execution time completed by all task till a specific instant of time) and the time quantum TQ. The MHR uses fixed time quantum TQ and gives preference for the shortest task to complete first. Initially TQ is fixed and the T_{EXE} is initialized as zero i.e., (T_{EXE}=0). After each time slice of execution the T_{EXE} is updated as T_{EXE}= T_{EXE}+TQ. The shortest jobs are given chance till the condition $T_{EXE} < T_{LARGE}$ is met then priority reverses to the task with largest execution time. Both MHM and MHR works like MH. The MHM and MHR make use of median and mean square root time quantum that dynamically varies after each round of Round Robin scheduling. The performance analysis of these shows the reduction in waiting time, turnaround time and the response time.

2: Dynamic Cloudlet scheduling in using Prioritized RR Algorithm

Sunitha Bansal Et al. proposed a heuristic task replication based RR scheduling algorithm for grid environment that is implemented using the state transition diagram [5]. The state diagram includes four states with two different queues i.e., the wait queue and the execution queue. The tasks from the waiting queue are sequenced into the execution queue in first come first served (FCFS) order and the scheduling of the task is done in RR order in the idle machines. The task that is mapped to the slow processing machines are replicated in the other high processing machines to speed up the execution and the resource utilization i.e., to improve the probability that the high processing machine completes the task before in hand than the slow processing machine to which the task is actually allocated. This gridlet replication can reduce the waiting time of the tasks and improve the execution time and the throughput. The performance is compared against normal RR algorithm and inferred fair improvement in throughput.

3: Performance Evaluation of Weighted RR in Grid

N. Krishnamoorthy Et al. had put forward a version of Weighted Round Robin (WRR) of scheduling for the grid environment [6]. The weight of each machine in this WRR is estimated using the number of hops and the bandwidth. The number of hops refers the number of supporting intermediate nodes to be bypassed in order reach

the processing node of the cluster. This minimizes the transfer latency and increase the transfer rate. The performance of this algorithm is estimated with traditional RR algorithm, the throughput has improved to a proportion of 15.96%.

4: Modified RR Algorithm for Resource Allocation

Pandaba Pradhan Et al. had proposed a modified version of Round Robin algorithm for resource allocation in the cloud environment [7]. The authors attempts to minimize the problem associated with RR algorithm i.e., time quantum selection by adopting dynamic time quantum in each round of RR execution. The time quantum is fixed by calculating the remaining execution time average of all active cloudlets at any specific round. The algorithm is implemented in MATLAB and the waiting time of the cloudlet is minimized.

5: Load Balancing using Improved Weighted Round Robin for Dependent Cloudlet

The author D.Chitra Devi Et al. proposed an improved version of Weighted Round Robin algorithm for dependent cloudlet scheduling [8]. Optimization in load balancing is attempted to achieved by considering the cloudlet length, resource configuration, interdependency among the cloudlets and by avoiding overloading by identifying the under loaded and overloaded virtual machines. The authors had scheduled the non pre-emptive cloudlets in the virtual machine. The performance is evaluated using different cases like scheduling heterogeneous cloudlet in the homogeneous VMs and scheduling the heterogeneous cloudlets in the heterogeneous VMs. The performance measure in terms of VM migration, completion time of cloudlets in timeshared and space shared policies is also estimated.

6: Adaptive Cloudlet Scheduling Algorithm using three Phase Optimization (ACS3O)

Mohan Lavanya Et al. proposed the ACS3O algorithm with the intention of minimizing the latency and achieving the uniform cloudlet distribution among the selected VMs [9]. This three phase approach starts with the VM categorization phase that estimates the maximum and minimum cloudlet size the virtual machine is capable of accepting. The next phase the load of all VMs is calculated along with its speed (MIPS). The adaptive phase finds the target VM status to load the cloudlet. If the target VM is busy then the threshold value of the VM is estimated, if it is below the overloaded threshold then the cloudlet is queued in the target VM. If the target VM is beyond the threshold, then the scheduler searches for the next VM for the cloudlet allocation. The algorithm results in reduced waiting time, improved VM utilization and the uniform cloudlet sharing.

7: Hybrid SJF and RR with Dynamic Variable Quantum Time Scheduling (SRDQ)

The SRDQ proposed by Samir Elmougy Et al. is a hybrid version of Shortest Job First (SJF) and RR algorithm with dynamic varying time quantum [10]. The proposed algorithm aims at minimizing the waiting time. The ready queue is divided into two sub-queues, one for the short length cloudlets (Q_1) and the other for the longest cloudlets (Q_2). The time quantum TQ set equal to the median of execution time all cloudlets. If the execution time of cloudlet is less than the TQ then place it inside Q_1 else place it in Q_2. Allocation of two cloudlets from Q_1 and one cloudlet from Q_2 is done in all iterations. The performance factors like throughput, waiting time, response time is optimized compared to SJF and RR algorithms. The proportion of meeting the deadline is also improved.

8: Scheduling Resources in Cloud using Threshold values at Host and Data center

Yatendra Sahu Et al. had plotted the compare and load balance algorithm [11]. This algorithm is a form of VM threshold base algorithm that optimizes the resource allocation by balancing the VM load and by reducing number of VM migrations. The algorithm commence by estimating the host and data center capacity limit and by setting the upper threshold limit for all resources. The allocation is made only if the resource threshold limit is suitable for allocation. The proposed algorithm results in improved resource utilization and equal spread of cloudlets.

9: **Credit based Scheduling Algorithm with Load Balancing (CBSALB)**

Narwal, Abhikriti Et al. [12] is proposed CBSALB algorithm to schedule the cloudlets among the VMs using the credit score of the cloudlets and the load balancing factor of the VMs. The credit score for the cloudlet is determined by considering various factors like cloudlet length, deadline, priority and cost. Honey Bee optimization algorithm is used for load balancing. The performance of the proposed algorithm is compared with CBSA proposed in [13]. The processing time, cost and makespan are improved compared to the CBSA algorithm.

10: Energy Efficient Dynamic Threshold based Load Balancing

Gupta, Shivani Et al. [14] has implemented the dynamic threshold based load balancing using virtual machine migration and server consolidation. The experiment is evaluated by setting upper and lower threshold limit for all VMs. The proposed methodology begins by estimating the load of all VMs and physical hosts. Then the threshold limit is estimated to find the under loaded and overloaded VMs and hosts. The appropriate VM is selected for migration and are placed in the right host of suitable capability and load limit. The authors inferred increased resource utilization and the number of migration have been diminished by increasing the CPU utilization.

III. Scheduling Criteria

Start Time (ST): The start time is defined as the very first time the cloudlet is allocated for execution in the VM [15]. If the cloudlet is scheduled in any VM with no load then the start time is zero. When the VM is loaded with other cloudlets already, then the start time of the cloudlet will be the completion time of the previous cloudlet that is in current execution. ST=0 if the VM has no load and S>1 when the VM already has some load in its buffer.

Execution Time: It is defined as the time taken by the cloudlet C_i to complete its execution in the specific instance of the virtual machine VM_j. It is expressed as the ratio between the size of the cloudlet in MI (million instructions) and the speed of the VM in MIPS (million instructions per second).

ET_i = (Size of cloudlet C_i in MI / Speed of virtual machine VM_j in MIPS) (1)

Completion Time (CT): Completion time or exit time is the summation of execution time of the cloudlet C_i and the completion time of the previously allocated cloudlet C_{i-1} in the same virtual machine VM_j.

CT_i = (Execution Time of the Cloudlet C_i + Completion Time of the previously allocated cloudlet C_{i-1} in the same VM) (2)

Arrival Time (AT): AT is the time at which the cloudlet C_i enters the ready queue of the virtual machine V_j.

Turnaround Time (TAT): TAT is the total time spent by the cloudlet C_i in the ready queue of the virtual machine VMj i.e., waiting time and its completion time.

TAT_i= Waiting Time of cloudlet C_i+ Completion Time of cloudlet C_i (3)

Waiting Time (WT): WT is the total time spent by the cloudlet C_i in the wait queue and the ready queue of the virtual machine VM_j for the execution.

WT_i= (Turnaround Time of the cloudlet C_i of the virtual machine VM_j – Execution Time of the cloudlet C_i on the virtual machine VM_j) (4)

Response Time (RT): RT is the time at which the cloudlet C_i in the ready queue is given the first chance for execution in the virtual machine VM_j.

RT_i=Time at which the cloudlet C_i gets the virtual Machine VM_j for the first time – Arrival time of the cloudlet C_i (5)

Makespan: The makespan is the time taken to complete all cloudlets in a cloudlet group. It is the summation of the execution time of all cloudlets and the transfer latency.

Throughput (TP): It specifies the number of cloudlet completed per unit time. The throughput estimates the efficiency of the scheduling algorithm.

IV. Review of Comparison Algorithms

Table 1: Review of RR algorithms taken for comparison and performance assessment

Algo. No.	Ref. No.	Title of Algorithm	Time Quantum Approximation (TQ)	Description
[A1]	[16]	Improved Shortest Remaining Burst Round Robin (ISRBRR)	TQ= SQ((SET/N) TQ= RET [C_{FRONT}]	Implements the square root approximation for the first round of TQ. Then it arranges the cloudlet based on Shortest Remaining Time First (SRTF) scheme. For the next round the TQ will be the RET of the cloudlet in the front end of the ready queue. SET: Summation of execution time of all N cloudlets. SQ: Square root function. N: Number of cloudlets. RET: Remaining Execution Time. RET [C_{FRONT}]: RET of the first front end cloudlet in the ready queue after SRTF arrangement.
[A2]	[17]	Half Life Variable Quantum Time Round Robin (HLVQTRR)	TQ=FLOOR(SET/N) TQ= RET [C_i]	Implements the mean approximation for the first round. Each cloudlet is given a TQ of almost half of its execution time in the first round. The next round each cloudlet C_i will be execute till the RET

				is completed.
[A3]	[18]	Improved Round Robin Varying Quantum (IRRVQ)	$TQ=RET[C_{FRONT}]$	Arrange cloudlets in ascending order of its execution time. The TQ for all other rounds is initialized as the RET of the first front end cloudlet in the ready queue of the VM.

A[4]	[19]	Dynamic Time Quantum Round Robin (DTQRR)	$TQ_1=ET[C_0]$
			If ($RET[C_{FRONT}]<=TQ_1$)
			Execute C_{FRONT}
			Else
			Execute $C_{FRONT+1}$
				TQ_1 is initialized with the first cloudlet execution time. The current cloudlet in execution is given a one more chance in the same round if the RET is less than or equal to TQ_1. Else switch to the next cloudlet by placing current cloudlet to the end of the ready queue.

V. Simulation Environment

The algorithm is implemented in the CloudSim environment. CloudSim is a Java based tool kit with rich set of built-in classes and libraries that supports the modelling of basic cloud environment. It allows the user to create number of Data centers, Host, Virtual Machines in a single stand alone machine. It renders support for performing the basic operations like data center creation, list of host creation, VM creation, cloudlet creation, VM scheduling, cloudlet scheduling etc. User can select the data center broker selection policy for placing their cloudlet in the right data center [L1]. Some of the CloudSim virtual machine and cloudlet parameters and are enumerated in Table 2 and Table 3.

Table 2: Virtual Machine Parameters

S.No.	Parameter	Description
1.	ID	Unique ID assigned for the VM.
2.	User_ID	Unique ID of Virtual Machine owner.
3.	MIPS	Capacity as Million Instructions per seconds.
4.	Number_of_Pes	Number of Processing Elements of CPU's.
5.	RAM	Size of input file in bytes before submission for processing.
6.	BW	Size of output file in bytes after processing.
7.	SIZE	Memory Size.
8.	VMM	Types of Virtual Machine Monitor.
9.	Cloudlet_Scheduler	Specifies the cloudlet scheduling policy.

Table 3: Cloudlet Parameters

S.No.	Parameter	Description
1.	Cloudlet_ID	A unique ID is assigned for the cloudlet and the cloudlet is referred with its Cloudlet_ID throughout its execution.
2.	Cloudlet_Length	Defines the number of executable instruction in the cloudlet that is measured in MI (million instructions).
3.	Pes_Number	Number of processing elements. Pes generally represents the CPU whose speed is measured in MIPS.
4.	Cloudlet_Filesize	Size of input file in bytes before it is submitted for processing.
5.	Cloudlet_Outputsize	Size of output file in bytes after processing.
6.	Utilization_ModelCPU	Return the utilization model used for cloudlet

7.	Utilization_ModelRAM	processing.
8.	Utilization_ModelBW	

VI. Proposed Algorithm

The time quantum selection in Round Robin algorithm is a big challenge. The inappropriate time slice may degrades the cloud system performance. When the time slice is larger the Round Robin algorithm behaves like First Come First Server (FCFS) algorithm, similarly for the smaller time slice the number of context switching is more. Context switching implies the method of, VM switching between one cloudlet to the other cloudlet. While switching, the status of the cloudlet is stored in VM buffer in order to restore or resume back for the cloudlet execution in near future. More context switching degrades the performance. So the proposed work makes an attempt to estimate an optimal time quantum that reduces the response time, waiting time, turnaround time and the number of context switching. The time quantum is allocated using mid and average remaining execution time approximation method. The Enhanced Time Quantum Round Robin Algorithm (ETQRRA) arranges the cloudlets in ascending order as per their execution time. When the number of cloudlets in the ready queue RQ[] of the VM is odd then the mid approximation method is used. If the number of cloudlet in the ready queue is even then the average remaining execution time approximation method is used. The pseudo code of the ETQRRA algorithm is elucidated below:

Enhanced Time Quantum Round Robin Algorithm (ETQRRA**)**

// Initialize the Ready Queue RQ[List of Cloudlets]
// WQ[]: Waiting Queue
// ETime [Execution Time of Cloudlets]
// N: Number of cloudlets in RQ[] at any given instant of time
// C_i: i^{th} Cloudlet in the ready queue RQ[]
// ETQ: Enhanced Time Quantum
// RET: Remaining Execution Time
Step 1: Check the status of Ready Queue RQ[]
if(RQ[]==EMPTY)
Assign the new cloudlets from WQ[] to the RQ[] in the FCFS order.
Step 2: Rearrange the RQ[] cloudlets in increasing order of its execution time.
Step 3:
While (RQ[]!=EMPTY) then
Repeat i=1 to N
if(N>1)
if(N%2==1) then
MID=FLOOR (N/2)
ETQ=ETime[MID]
Else
ETQ=FLOOR ($\sum_{k=0}^{N}$ ETime/N) // Average of RET
End if
Else
ETQ=ETime[Front]
End if
ExeProcess(C_i, ETQ) // Executing Cloudlet C_i for time quantum ETQ
End Repeat
Step 4: Append the newly arrived cloudlet to the end of the RQ[] and arrange the RQ[] in increasing order of their remaining execution time (RET).
Step 5: Calculate the new value of N i.e., the number of cloudlets in RQ[]
Step 6: Repeat Step 1 to 5 until no cloudlet is left
Step 7: Estimate the response time, waiting time, turnaround time and the number of context switching.
Step 8: End

Procedure ExeProcess (Cloudlet C_i, Time ETQ)
Execute the Cloudlet C_i the for time quantum ETQ

RET_i=ETime[i]-EQT // Calculate RET of cloudlet C_i
if(RET==0) then
Remove Cloudlet C_i from RQ[]
Else
ETime[i] =RET_i
End ExeProcess

VII. Results and Discussion

The simulation is done by taking five cloudlets of varying size and execution time. The static allocation is done using the aforementioned algorithms. Table 4. depicts the list of cloudlets with its ID and its execution time. Table 5. sorts the cloudlet based on their execution time.

Table 4: Original Cloudlet Table

S. No.	Cloudlet ID	Execution Time (m-sec)
1.	P1	10
2.	P2	12
3.	P3	23
4.	P4	20
5.	P5	15

Table 5: Sorted Cloudlet Table

S. No.	Cloudlet ID	Execution Time (m-sec)
1.	P1	10
2.	P2	12
3.	P5	15
4.	P4	20
5.	P3	23

Time Quantum for Iteration one: Odd number of cloudlets where N=5
MID= Floor (5/2)=2
ETQ= ETime[C]=15 m-sec.
Time Quantum for Iteration two: Even number of cloudlets where N=2
ETQ= FLOOR ($\sum_{k=0}^{N}$ ETime/N)
ETQ=6 m-sec.
Time Quantum for Iteration three: Odd number of cloudlets where N=1
ETQ=ETime [Front]=2 m-sec.
Gantt chart cloudlet execution:

The iteration 1 the time quantum is fixed to be 10 m-sec. So P_1 executes for 10 m-sec followed by context switching from P_2, P_5, P_4, and P_3. Now P_1, P_2, P_5 exits the queue as its remaining execution time is zero. Then for the next iteration the time quantum is fixed to be 6 m-sec, P_4 exits the queue after executing for 5 m-sec. Then the P_3 continues its execution since there is no other cloudlet is left to be executed. Figure 1. depicts the Gantt chart execution sequence of the cloudlets.

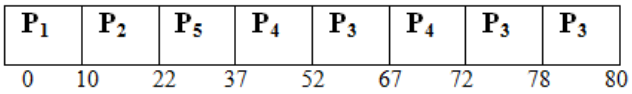

Figure 1. Gantt chart of ETQRRA Algorithm

Table 6. comprises of the average response time, waiting time, average turnaround time and the number of context switch incurred by CRR, ISRBRR, HLVQTRR, IRRVQ, DTQRR and ETQRRA algorithms.

Table 6: Performance Analysis of Algorithms

Algorithm	Average Response Time	Average Waiting Time	Average Turn Around Time	No. of Context Switch
CRR	29.6	32.4	48.4	7
ISRBRR [2]	8	28.6	44.6	8
HLVQTRR [3]	24.4	61.2	76.2	22
IRRVQ [4]	20	40.2	56.2	15
DTQRR [5]	20	27.2	43.2	10
ETQRRA	24.2	28.2	44.2	8

The statistical graph in Figure 2, Figure 3, Figure 4 and Figure 5 compares the performance of aforementioned algorithms in terms of mean response time, waiting time, turnaround time and the number of context switch.

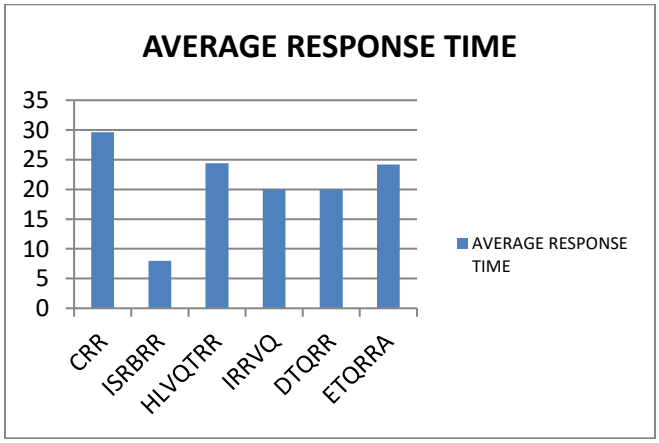

Figure 2. Analysis of algorithms with respect to response time

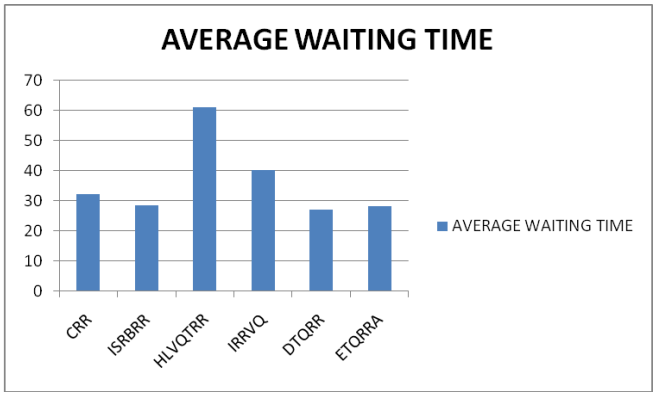

Figure 3. Analysis of algorithms with respect to wait time

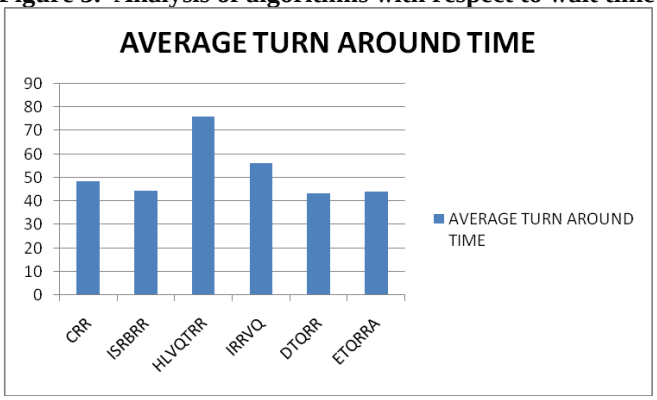

Figure 4. Analysis of algorithms with respect to turnaround time

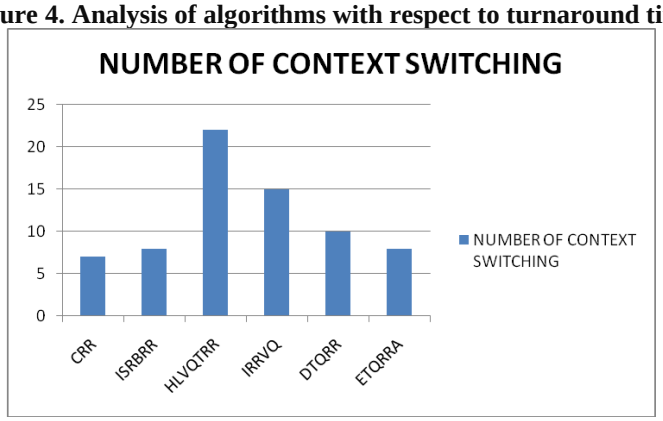

Figure 5. Analysis of algorithms with respect to number of context switching

VIII. Conclusion

This paper provides an overview of various key concepts of cloud computing like virtualization, scheduling and load balancing. The comparative analysis of the proposed ETQRRA algorithm with the other versions of the existing Round Robin algorithms like CRR, ISRBRR, HLVQTRR, IRRVQ and DTQRR shows that both DTQRR and ETQRRA provides a better result than the other algorithms. The ETQRRA and DTQRR algorithms results in almost equal waiting and turnaround time compare to the other algorithms. In turn the ETQRRA had also resulted in decreased number of context switching. The limitation of this algorithm is that, it schedules the cloudlets statically based on the execution time of the cloudlets alone. Further refinements can done by considering the other factors of the cloudlets and the VMs like cloudlet size (MI), number of processing elements PEs in the VM, bandwidth (Mbps), memory and speed (MIPS) of the VM.

References

1. Savu, Laura, "Cloud computing: Deployment models, delivery models, risks and research challenges", International Conference on Computer and Management (CAMAN), IEEE, 2011.
2. Ding, Wei Min, Benjamin Ghansah, and Yan Yan Wu, "Research on the Virtualization Technology in Cloud Computing Environment", International Journal of Engineering Research in Africa. Vol.21. Trans Tech Publications Ltd, 2016.
3. Nora Almezeini and Alaaeldin Hafez, "Review on Scheduling in Cloud Computing", International Journal of Computer Science and Network Security, VOL.18 No.2, February 2018
4. Shah, Syed Nasir Mehmood, Ahmad Kamil Bin Mahmood, and Alan Oxley, "Dynamic Multilevel Hybrid Scheduling Algorithms for Grid Computing", Procedia Computer Science 4 (2011): 402-411.
5. Bansal, Sunita, Bhavik Kothari, and Chittaranjan Hota, "Dynamic Cloudlet-Scheduling in Grid Computing using Prioritized Round Robin Algorithm", International Journal of Computer Science Issues (IJCSI) 8.2 (2011): 472.
6. Krishnamoorthy N, R. Asokan, and S. Sangeetha, "Performance Evaluation of Weighted Round Robin Grid Scheduling", International Journal of Computer Applications 68.13 (2013).
7. Pradhan, Pandaba, Prafulla Ku Behera, and B. N. B. Ray, "Modified Round Robin Algorithm for Resource Allocation in Cloud Computing", Procedia Computer Science 85 (2016): 878-890.
8. D.Chitra Devi and V. Rhymend Uthariaraj, "Load balancing in Cloud Computing Environment using Improved Weighted Round Robin Algorithm for Nonpreemptive dependent cloudlets", The scientific world journal 2016 (2016).
9. Lavanya, Mohan, B. Santhi, and Sivasankaran Saravanan, "Adaptive Cloudlet Scheduling Algorithm Using Three Phase Optimization Technique", International Conference on Advances of Science and Technology, Springer, Cham, 2018.
10. Elmougy, Samir, Shahenda Sarhan and Manar Joundy, "A novel hybrid of Shortest job first and round Robin with Dynamic Variable Quantum Time Cloudlet Scheduling Technique", Journal of Cloud computing 6.1 (2017): 1-12.
11. Yatendra Sahu, Neha Agrawal, "Scheduling Resources in Cloud using Threshold Values at Host and Data Center Level", International Journal of Computer Science and Information Technologies, Vol 6 (6),2015, 4982-4986.
12. A. Narwal, S. Dhingra, "Credit based scheduling with load balancing in cloud environment", International Journal of Advanced Trends in Computer Science and Engineering, 9 (2) (2020) 1121–1127.
13. Narwal, Abhikriti, and Sunita Dhingra, "Analysis of credit-based scheduling algorithms in the cloud computing framework", Materials Today: Proceedings (2020).
14. Gupta, Shivani, Damodar Tiwari, and Shailendra Singh. "Energy efficient dynamic threshold based load balancing technique in cloud computing environment." Int. J. Computer Science and Information. Technology, 6.2 (2015): 1023-1026.
15. Chatterjee, Tamojit Et al. "Design and Implementation of an Improved Data Center Broker Policy to Improve the QoS of a Cloud", Proceedings of the Fifth International Conference on Innovations in Bio-Inspired Computing and Applications. Springer, Cham, 2014.
16. P.Surendra Varma, "Improved Shortest Remaining Burst Round Robin (ISRBRR) using RMS as its Time Quantum", International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), Vol.1, Issue8, (2012).
17. Simon Ashiru , Salleh Abdullahi , Sahalu Junaidu, "Half Life Variable Quantum Time Round Robin (HLVQTRR)", International Journal of Computer Science and Information Technologies, Vol.5(6) (2014).
18. Manish Kumar Mishra and Dr. Faizur Rashid, "An Improved Round Robin CPU Scheduling Algorithm with Varying Time Quantum", International Journal of Computer Science, Engineering and Applications (IJCSEA), Vol.4, No (2014).
19. Yosef Berhanu, Abebe Alemu, Manish Kumar Mishra, "Dynamic Time Quantum based Round Robin CPU Scheduling Algorithm", International Journal of Computer Applications, Vol.167 – No.13 (2017).
20. https://www.cloudsimtutorials.online/