SOCA
DOI 10.1007/s11761-016-0196-3

ORIGINAL RESEARCH PAPER

Design and analysis of an efficient QoS improvement policy in cloud computing

Sourav Banerjee[1] · **Mainak Adhikari**[2] · **Utpal Biswas**[3]

Received: 2 May 2015 / Revised: 28 April 2016 / Accepted: 18 July 2016

Abstract Cloud computing offers the proficiency to use computing and storage resources on a metered basis and reduces the investments in Information Technology domain. This paper highlights a major research issue, i.e., providing good quality of service (QoS) to the cloud users. The QoS is associated with several parameters such as completion time, response time, turnaround time (TAT), waiting time (WT), bandwidth. A new cloudlet scheduling algorithm—improved round robin cloudlet scheduling algorithm—has been proposed which improves the TAT, WT and number of context switching. It enhances the resource utilization. The experimental results are obtained by CloudSim toolkit extending few base classes and compared by classical round robin algorithm.

Keywords Cloud user (CU) · Quality of service (QoS) · Improved round robin cloudlet scheduling algorithm (IRRCSA) · Round robin algorithm (RRA)

✉ Sourav Banerjee
mr.sourav.banerjee@ieee.org

Mainak Adhikari
onlinemainak@yahoo.com

Utpal Biswas
utpal01in@yahoo.com

1 Kalyani Government Engineering College, Kalyani, Nadia 741235, India

2 IMPS College of Engineering and Technology, Malda 732103, India

3 University of Kalyani, Kalyani, Nadia 741235, India

1 Introduction

Over the years, distributed environments have evolved from shared community platform to utility-based models; the latest of these being cloud computing [1–3]. The cloud computing enables the delivery of IT resources over the Internet and follows an *on-demand service model* where the users are charged based on their consumption. There are various types of Cloud providers into hierarchy of as-a-service terms [4,5]—software as a service (SaaS), platform as a service (PaaS), infrastructure as a service (IaaS), database as a service (DaaS), identity as a service (IdaaS), etc. Moreover, they offer the flexibility, elasticity and scalability to acquire or release resources with varying configuration to best suit the requirements of an application. Even though this empowers the cloud users (CUs) [6] and gives the registered users more control over the resources. It also dictates the development of innovative scheduling techniques so that the distributed resources are efficiently utilized. The cloud computing [7,8] is a type of parallel and distributed system consisting of collection of large-scale heterogeneous interconnected and virtualized [9,10] computers that are dynamically provisioned and presented as one or more unified computing resources established through negotiation between the service providers and customers.

In this domain, the background activities like virtual machine (VM) [10,11] allocation, load sharing, load balancing [3,12], cloudlet migration, and distributed shared memory access are completely abstracted from the user's purview. Here, the end users or the customers can access the cloud-based applications [3,4] as well as infrastructure through logging in to a cloud interface. To make the cloud services proficient in that environment, one of its challenges is to provide an efficient cloudlet scheduling policy. The cloudlet scheduling policy [11,13] plays a vital role to improve the overall system performance minimizing the turnaround time, waiting time and context switching. A proper scheduling policy may eventually lead to improve the QoS [14] of the overall system.

1.1 Our contribution

In the domain of cloud computing [15], several research works are going on. There are many traditional methods which are used to highlight the parameters involved in QoS of cloud. Based on the literature survey, it is found that little amount of research work has been done to emphasize the QoS parameter evaluation. In this paper, a new cloudlet scheduling algorithm has been proposed and simulated in the famous CloudSim 3.0.3 [16]. The obtained result sets are compared by RRA. In comparison with RRA, the proposed IRRCSA provides better turnaround time, waiting time and context switching of the cloudlets allotted to the VMs and VMs present in the host in DC. Hence, the system utilization has been improved also. So many research works have been undertaken, based on scheduling techniques with various network scenarios and combinations of service classes [17–19]. A detail description and justification with proper analysis is sited in Sects. 4 and 5.

1.2 Organization

The rest of the paper is organized as follows—Sect. 2 describes the different related works regarding existing cloudlet scheduling algorithms. In Sect. 3, the proposed work has been emphasized with algorithm and flowchart. In Sect. 4, the experimental result of the proposed scheduling algorithm—IRRCSA—is described. Section 5 presents the comparison result and simulated graphs to illustrate the prominence of this proposed algorithm over some existing algorithms. Finally, Sect. 6 concludes and discusses future scope of the proposed work.

2 Related work

2.1 CloudSim toolkit

Several grid simulators [20–22], such as GridSim, SimGrid and GangSim, are capable of modeling and simulating the grid application in a distributed environment, but fail to support the infrastructure and application-level requirements arising from cloud computing paradigm [23]. A cloud infrastructure modeling and simulation toolkits must support real-time trading of services between customers and providers. The open-source CloudSim framework [6,24] shown in Fig. 1 is developed on GridSim toolkit that offers support for economic-driven resource management and application scheduling simulation. It provides users a

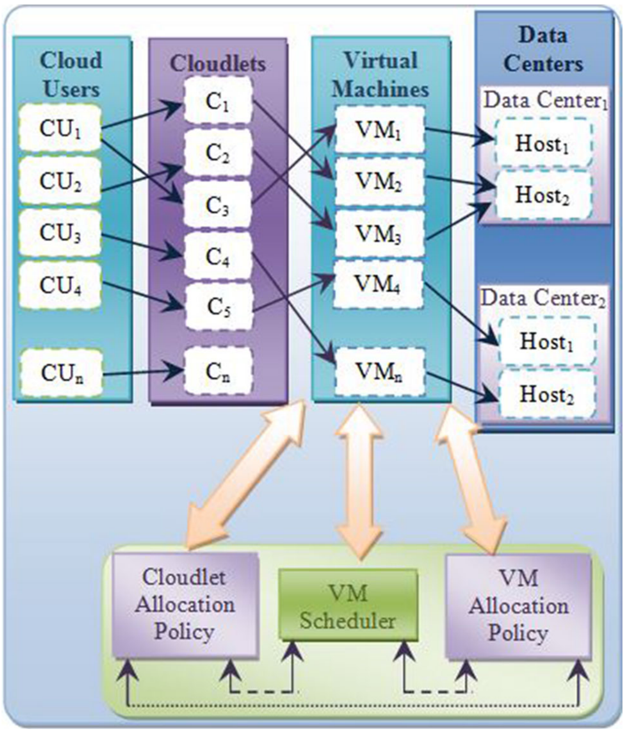

Fig. 1 CloudSim work style

series of extended entities and methods. In addition, it helps users to analyze their own scheduling and allocation strategy at different levels including modification of module deployment techniques and conduct related performance testing by expanding few interfaces. The present study aims at expanding CloudSim by utilizing the broker policy. The data center broker policy is a decision-making procedure which makes the best match between cloudlets and VMs. The modules of CloudSim toolkit which are relevant to our research are as follows.

- *VM scheduler* VM scheduler is an abstract class implemented by a host component, represents the policies (space-shared, time-shared) required for scheduling VMs to PE (Processing Element) [12,25].
- *VM allocation policy*VM allocation policy [12,25] is used to select available host in a data center, which meets the memory, storage and availability requirement for a VM deployment.
- *Cloudlet scheduling* The cloudlet scheduler selects the cloudlets from the local queue (LQ) of each VM and allocates them in the available VM

2.2 Scheduling algorithm

There are many different types of cloudlet scheduling [17,24] algorithms present, but one them has been chosen in this study.

Round robin algorithm (*RRA*) [17,18]: In this policy, a small unit of time is defined, which is known as time quantum (TQ). The execution time [26] of a cloudlet is decreased by the TQ. If the execution time of a cloudlet is less than or equal to the TQ, that is allowed to continue till normal termination. If the execution time is greater than TQ, the cloudlet is preempted and added to the tail of the LQ. Then, the next cloudlets in the LQ start running. It will continue until the cloudlet is completed.

For example, there are three cloudlets with their execution time $C_0 = 25$, $C_1 = 10$ and $C_2 = 12$ and the TQ is 5.

C_0	C_1	C_2	C_0	C_1	C_2	C_0	C_2	C_0
0	5	15	20	25	30	35	37	47

Number of context switch (CS) is 8 (Tables 1 and 2).

Disadvantages:

(i) The average waiting time and turnaround time is maximum in comparison with the proposed IRRCSA.
(ii) Number of CS is larger in comparison with IRRCSA.

The disadvantages are justified in Sects. 4 and 5.

3 Proposed work

The proposed IRRCSA has overcome the disadvantages of the above scheduling algorithm. In this proposed work, all the VMs are stored in *vm_list* [11] array with their corresponding million instructions per second (MIPS) and also a batch of cloudlets is stored in the *cloudlet_list* [11] array with their corresponding million instructions (MI). Initially, the cloudlets are assigned to the LQ(s) of the VM(s), following the circular first come first serve (FCFS) manner. This cloudlet allocation policy continues till all the cloudlets in the global queue (GQ) are allotted into the LQ(s) of the VM(s). Then, the expected execution time (ET) of each cloudlet is calculated according to the Eq. 1. The ET of a cloudlet is defined as the ratio of MI (Million Instruction) of the cloudlet to the MIPS of the allotted VM. Then, sort the cloudlets in ascending order according to their ET. After sorting the cloudlets, the data center broker (DCB) sets a time quantum (TQ) and starts executing the cloudlets. The execution of the cloudlet continues till the TQ expires, and the scheduler checks the remaining execution time (RET) of the cloudlet concurrently following Eq. 2. If the RET of that cloudlet is less than the expected execution time among the remaining cloudlets in the LQ of that VM, then it is allowed to continue its execution. Otherwise, the cloudlet is switched and the next cloudlet from the *cloudlet_list* starts its execution. This process continues till the LQ becomes empty. The model is mentioned in Fig. 2.

Table 1 Average waiting time using RRA

cld_id	Waiting time
0	22
1	15
2	25
Average waiting time	20.7

Table 2 Average turnaround time using RRA

cld_id	Turnaround time
0	47
1	25
2	37
Average turnaround time	36.33

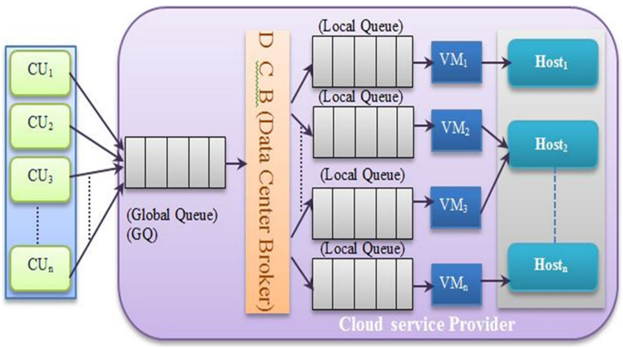

Fig. 2 Proposed model for IRRCSA

$$\text{Execution time}_j\ (\text{ET}_j) = \text{MI of cloudlet}_j/\text{MIPS of VM}_i. \quad (1)$$

$$\text{Remaining execution time}_j = \text{Execution time}_j - \text{TQ}. \quad (2)$$

3.1 Proposed model

Initially, the CUs [11] send batches of cloudlets to the CSP [11] that are stored in global queue (GQ). Then, the data center broker (DCB) [11] selects the VM from *vm_list.* Then, the VMs are allotted to the host in a data center (DC) [11] according to VM allocation policy [11]. The DCB module assigns the cloudlets to the VMs according to proposed policy.

3.2 Scheduling criteria

(i) ***VM utilization*** [16]: It is the average fraction of time. It ranges from 0 to 100 %. It keeps the VM as busy as possible.
(ii) ***Throughput*** [16]: Throughput is defined as the rate of the cloudlets completed per unit of time (number of cloudlets/unit time).

(iii) ***Context switching*** [16,27]: It is the process of storing and restoring the state of cloudlets so that execution can be resumed from the same point at a later time.

(iv) ***Fairness*** [16]: Each of the cloudlets should have the fair [17] share of VM. All tasks must get their chance at VM.

(v) ***Response time*** [16]: It gets first response on the VM to start its cloudlet execution.

(vi) ***Waiting time*** [16]: It is an amount of time a cloudlet needs to wait in ready queue. In RRA and IRRCSA scheduling algorithm, waiting time is referred to as the time for which the cloudlet is ready to execute but cannot execute by the VM.

(vii) ***Turnaround time*** [16]: It means the amount of time taken to execute a requested cloudlet by the VMs. In RRA and IRRCSA scheduling, turnaround time is defined as the total time taken by the cloudlet between the submission of a cloudlet for execution and the return of the complete output.

3.3 IRRCSA algorithm

```
1.    Initially creates the VMs and sort them in ascending order according to their MIPS value and store them in vm_list.
2.    The new batch of cloudlets are stored in cloudlet_list.
3.    The Cloudlets are allotted to the VMs following First Come First Serve manner until all the cloudlets are assigned to the VMs.
4.    Assign Time Quantum (TQ) of the cloudlets.
5.    For each VM present in the vm_list
While (Local Queue! = NULL)
  {
        for I = 1 to n
              {
              for j = 1 to k
              {
           Execution time_j = MI_j / MIPS_i;
Remaining Execution Time_j = Execution time_j – TQ;
if (Remaining Execution time_j <= TQ)
{
// the current executing cloudlet continue its execution
  }
     else
  {
// the currently executing cloudlets context switch and the next cloudlet from the LQ of that VM starts it's execution.
   }
}
// the process is continued until all the allotted cloudlets of different VMs present in vm_list did not complete their execution.
        }
    }
```

3.4 Explanation of the IRRSCA algorithm

Initially, the *vm_list* array is prepared by creating the VMs and arranging them in ascending order following their MIPS. The cloudlets are created and stored in cloudlet_list. Here, "n" is the number of VMs present in the *vm_list* and "k" is the number of cloudlets allotted to each VM.

Each VM contains a LQ where the cloudlets are allocated following FCFS manner. Then, TQ is assigned for fare share execution of each cloudlets. The execution of the cloudlet continues till the TQ expires, and the scheduler checks the

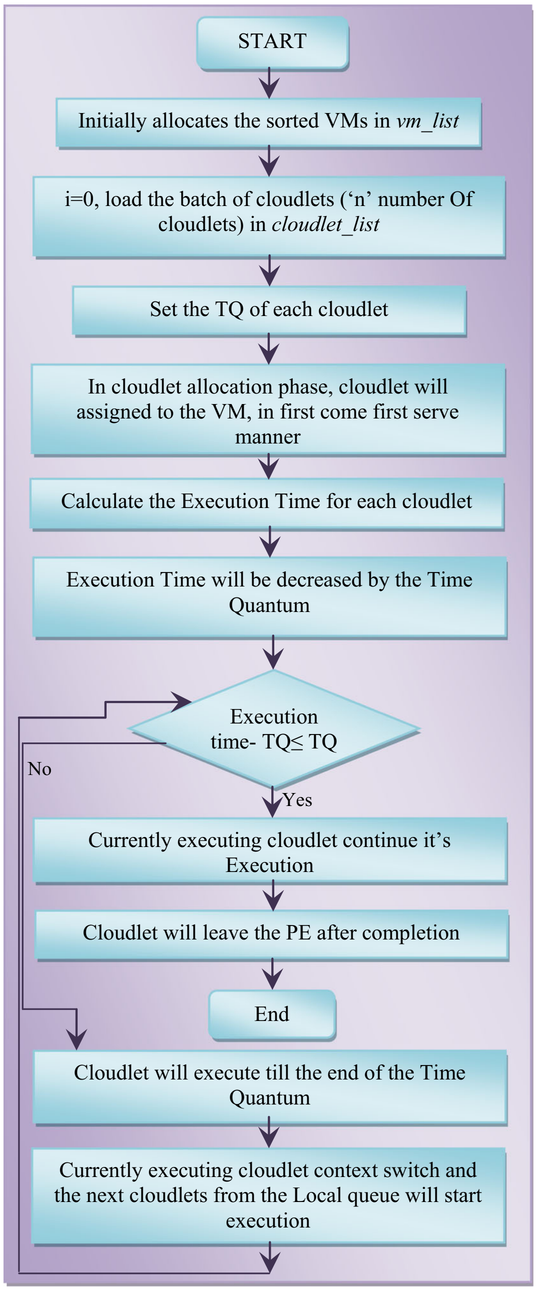
START
Initially allocates the sorted VMs in vm_list
i=0, load the batch of cloudlets ('n' number Of cloudlets) in cloudlet_list
Set the TQ of each cloudlet
In cloudlet allocation phase, cloudlet will assigned to the VM, in first come first serve manner
Calculate the Execution Time for each cloudlet
Execution Time will be decreased by the Time Quantum
Execution time- TQ≤ TQ
No
Yes
Currently executing cloudlet continue it's Execution
Cloudlet will leave the PE after completion
End
Cloudlet will execute till the end of the Time Quantum
Currently executing cloudlet context switch and the next cloudlets from the Local queue will start execution

Fig. 3 Flowchart of proposed IRRCSA

RET of the cloudlet concurrently following Eq. 2. If the RET of that cloudlet is less than the expected execution time among the remaining cloudlets in the LQ of that VM, then it is allowed to continue its execution. Otherwise, the cloudlet is switched and the next cloudlet from the *cloudlet_list* starts its execution. This process continues till the LQ becomes empty.

3.5 Flowchart of proposed IRRCSA algorithm

Figure 3 shows the flowchart of the proposed cloudlet scheduling algorithm—IRRCSA.

4 Experimental result

The proposed cloudlet scheduling algorithm has been described and analyzed using a suitable example. Due to space constraint, ten cloudlets and three VMs have been considered in this experiment. We have considered waiting time, turnaround time and context switching to evaluate the performance of the proposed algorithm in this paper.

Tables 3 and 4 show ten cloudlets with their size in millions of instruction (MI) and four VMs with their processing power in MIPS, respectively. The cloudlet id is represented as cld_id and VM id is represented as vm_id.

Table 5 plays a significant role in describing the cloudlet to the VM mapping as well as execution time for each cloudlet. It shows the cloudlets are allotted to the corresponding VM following the proposed IRRCSA algorithm.

Table 3 Properties of cloudlets

cld_id	MI
0	9000
1	16,000
2	11,000
3	6000
4	15,000
5	8000
6	12,000
7	17,000
8	10,000
9	7000

Table 4 Properties of VM

vm_id	MIPS
0	300
1	600
2	400
3	500

The execution time of allotted cloudlets is shown separately in Tables 6, 7, 8 and 9. Without calculating the execution time the RET cannot be estimated.

The VM-specific improvement in waiting time and turnaround time has been clearly mentioned from Tables 10, 11, 12, 13, 14, 15, 16 and 17. The rate of improvement of parameters has been mention.

Tables 10 and 11 show the improvement of waiting time and turnaround time of IRRCSA over RRA, respectively, in VM_0. The average waiting time and turnaround time has been mentioned to justify the improvement.

Table 5 Cloudlet to VM map table with execution time

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 6 Allotted cloudlet to VM_0 with their execution time

vm_id	MIPS	cld_id	MI	ET
0	300	0	9000	30
0	300	4	15,000	50
0	300	8	10,000	33

Table 7 Allotted cloudlet to VM_1 with their execution time

vm_id	MIPS	cld_ id	MI	ET
1	600	1	16,000	27
1	600	5	8000	13
1	600	9	7000	12

Table 8 Allotted cloudlet to VM_2 with their execution time

vm_id	MIPS	cld_id	MI	ET
2	400	2	11,000	28
2	400	6	12,000	30

Table 9 Allotted cloudlet to VM_3 with their execution time

vm_id	MIPS	cld_id	MI	ET
3	500	3	6000	12
3	500	7	17,000	34

Table 10 Waiting time for RRA and IRRCSA

vm_id	cld_id	RRA	IRRCSA
0	0	50	40
0	4	63	58
0	8	65	55
Average waiting time		59.33	51
Rate of improvement		**14.04 %**	

Bold signifies rate of improvement

Table 11 Turnaround time for RRA and IRRCSA

vm_id	cld_id	RRA	IRRCSA
0	0	80	70
0	4	113	113
0	8	98	93
Average turnaround time		97	92
Rate of improvement		**5.15 %**	

Bold signifies rate of improvement

Table 12 Waiting time for RRA and IRRCSA

vm_id	cld_id	RRA	IRRCSA
1	1	25	25
1	5	25	15
1	9	28	23
Average waiting time		26	21
Rate of Improvement		**19.23 %**	

Bold signifies rate of improvement

Table 13 Turnaround time for RRA and IRRCSA

vm_id	cld_id	RRA	IRRCSA
1	1	52	52
1	5	38	28
1	9	40	35
Average turnaround time		43.33	38.33
Rate of improvement		**11.53 %**	

Bold signifies rate of improvement

Table 14 Waiting time for RRA and IRRCSA

vm_id	cld_id	RRA	IRRCSA
2	2	25	20
2	6	28	28
Average waiting time		26.5	24
Rate of improvement		**9.43 %**	

Bold signifies rate of improvement

Tables 12 and 13 show the improvement of waiting time and turnaround time of IRRCSA over RRA, respectively, in VM_1.

Tables 14 and 15 show the improvement of waiting time and turnaround time of IRRCSA over RRA, respectively, in VM_2.

Table 15 Turnaround time for RRA and IRRCSA

vm_id	cld_id	RRA	IRRCSA
2	2	53	48
2	6	58	58
Average turnaround time		55.5	53
Rate of improvement		**4.50 %**	

Bold signifies rate of improvement

Table 16 Waiting time for RRA and IRRCSA

vm_id	cld_id	RRA	IRRCSA
3	3	10	5
3	7	12	12
Average waiting time		11	8.5
Rate of improvement		**22.72 %**	

Bold signifies rate of improvement

Table 17 Turnaround time for RRA and IRRCSA

vm_id	cld_id	RRA	IRRCSA
3	3	22	17
3	7	46	46
Average turnaround time		34	31.5
Rate of improvement		**7.35 %**	

Bold signifies rate of improvement

Table 18 Summery table for performance gain

Performance measure	VM	Rate of improvement (%)
Waiting time	VM_0	14.04
Turnaround time	VM_0	5.15
Waiting time	VM_1	19.23
Turnaround time	VM_1	11.53
Waiting time	VM_2	9.43
Turnaround time	VM_2	4.50
Waiting time	VM_3	22.72
Turnaround time	VM_3	7.35

Tables 16 and Table 17 show the improvement of waiting time and turnaround time of IRRCSA over RRA, respectively, in VM_3. The rate of improvement has been shown in Table 18.

5 Comparison result and simulated graph

This section deals with analyzing the improvement of the QoS in association with three parameters, such as waiting time, turnaround time and context switching. The performance of the proposed IRRCSA is compared and analyzed by RRA. The simulated results are evaluated and analyzed in several aspects. The performance was measured by allocating cloudlets to the corresponding VMs present in the host. The improvement of waiting time, turnaround time and context switching is explained with the help of the simulated graphs from Figures 4, 5, 6, 7, 8, 9, 10, 11, 12 and 13. The improvement in the result using the proposed IRRCSA indicates the enhancement of the QoS of a cloud.

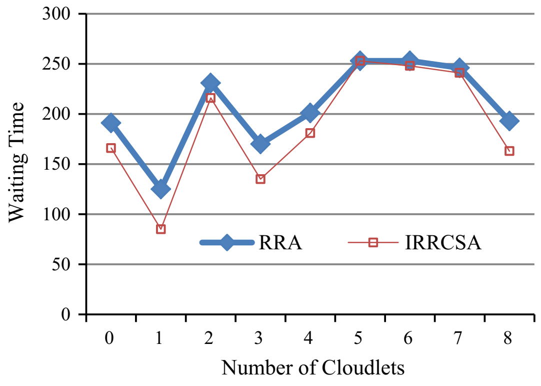

Fig. 4 Waiting time comparison: RRA vs. RRCSA in VM_0

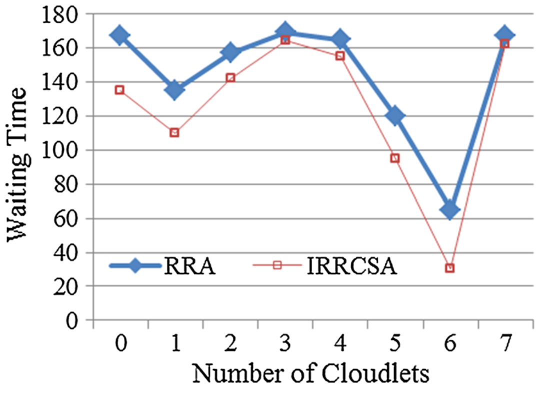

Fig. 5 Waiting time comparison: RRA vs. IRRCSA in VM_1

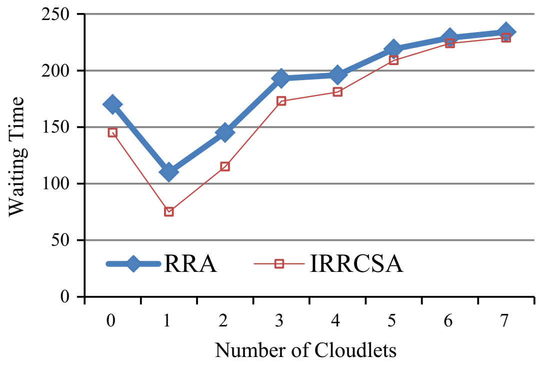

Fig. 6 Waiting time comparison: RRA vs. IRRCSA in VM_2

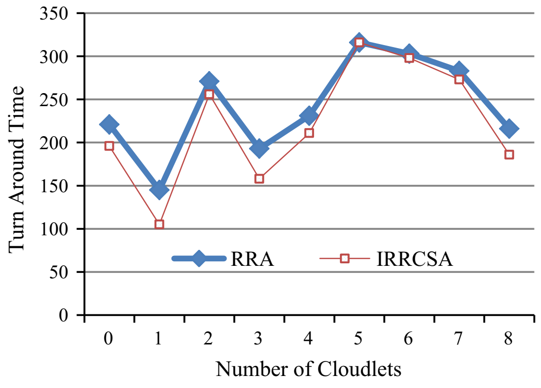

Fig. 7 Turnaround time comparison: RRA vs. IRRCSA in VM_0

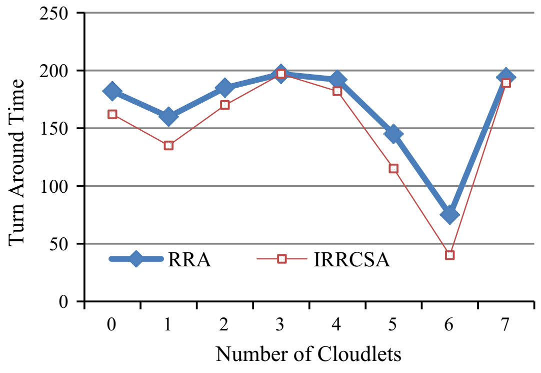

Fig. 8 Turnaround time comparison: RRA vs. IRRCSA in VM_1

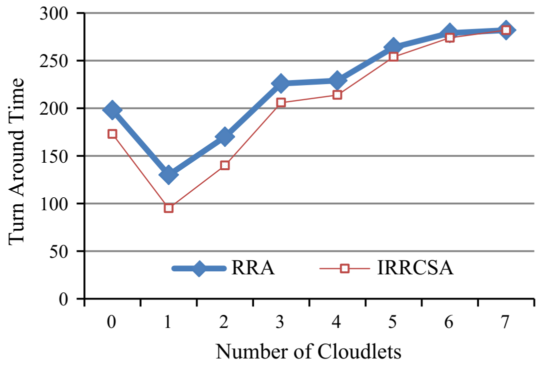

Fig. 9 Turnaround time comparison: RRA vs. IRRCSA in VM_2

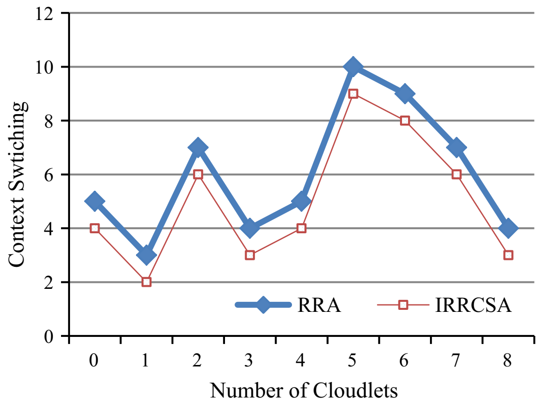

Fig. 10 Context switching comparison: RRA vs. IRRCSA in VM_0

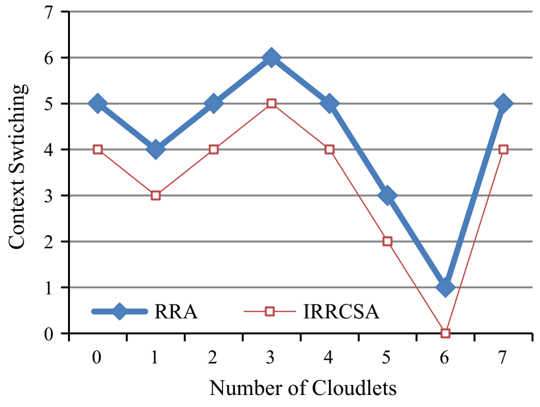

Fig. 11 Context switching comparison: RRA vs. IRRCSA in VM_1

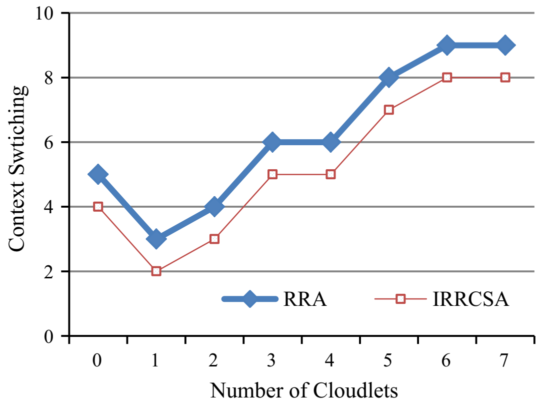

Fig. 12 Context switching comparison: RRA vs. IRRCSA in VM_2

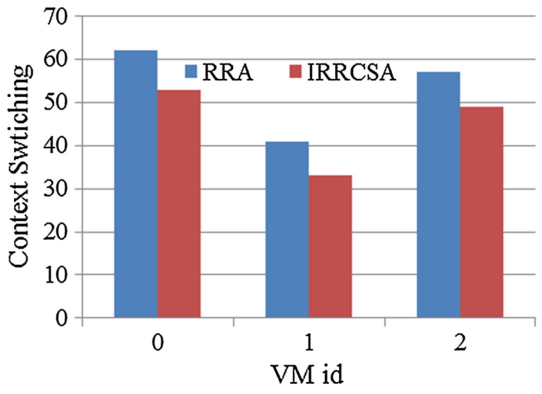

Fig. 13 Context switching of comparison: RRA vs. IRRCSA

Waiting time: In Figs. 4, 5 6 the improvement of waiting time is presented.

Turnaround time In Figs. 7, 8 9 shows the improvement of turnaround time.

Context switching In Figs. 10, 11 and 12 the improvement of context switching of the cloudlets is presented.

Figure 13 shows the improvement of CS of the VMs.

6 Conclusion and future work

The proposed work provides better turnaround time and waiting time and returns reduced number of context switching. The basic idea of the proposed "IRRCSA" mechanism is to leverage the strengths of round robin algorithm. This modification supports to minimize several parameters associated with QoS.

This proposed technique does not consider the fault tolerance issues. We will extend this work to include few issues like fault tolerance and scalability in the next work.

References

1. Xiong K, Perros H (2009) Service performance and analysis in cloud computing. 2009 Congress on services - I, Los Angeles, CA, pp 693–700. doi:10.1109/SERVICES-I.2009.121
2. Sotomayor B, Montero RS, Llorente IM, Foster I (2009) Virtual infrastructure management in private and hybrid clouds. IEEE Internet Comput 13(5):14–22. doi:10.1109/MIC.2009.119
3. Adhikari M, Banerjee S, Biswas U (2012) Smart task assignment model for cloud service provider. Spec Issue Int J Comput Appl (0975–8887) Adv Comput Commun Technol HPC Appl-ACCTHPCA, June 2012
4. Lei X, Zhe X, Shaowu M, Xiongyan T (2009) Cloud computing and services platform construction of telecom operator. In: 2nd IEEE international conference on broadband network & multimedia technology, IC-BNMT '09, Beijing, pp 864–867. doi:10.1109/ICBNMT.2009.5347793
5. Banerjee S, Adhikari M, Biswas U (2014) Development of a smart job allocation model for a cloud service provider. In: 2014 2nd international conference on business and information management (ICBIM), Durgapur, pp 114–119. doi:10.1109/ICBIM.2014.6970946
6. Calheiros RN, Ranjan R, De Rose CAF, Buyya R (2009) CloudSim: a novel framework for modeling and simulation of cloud computing infrastructures and services. arXiv:0903.2525
7. Armbrust M, Fox A, Griffith R, Joseph AD, Katz R, Konwinski A, Lee G, Patterson D, Rabkin A, Stoica I, Zaharia M (2010) Above the clouds: a berkeley view of cloud computing. Technical report No. UCB/EECS-2009-28. University of California at Berkley. http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html
8. Aymerich FM, Fenu G, Surcis S (2008) An approach to a cloud computing network. In: First international conference on the applications of digital information and web technologies, ICADIWT 2008, Ostrava, pp 113–118. doi:10.1109/ICADIWT.2008.4664329
9. Buyya R, Ranjan R, Calheiro RN (2009) Modeling and simulation of scalable cloud computing environments and the CloudSim toolkit: challenges and opportunities. In: Proceedings of the high performance computing and simulation conference, HPCS'09, IEEE, pp 1–11
10. White Paper-VMware Infrastructure Architecture Overview. VMware, CA
11. Bhatia W, Buyya R, Ranjan R (2010) CloudAnalyst: a CloudSim based visualmodeller for analysing cloud computing environments and applications. In: 2010 24th IEEE international conference on advanced information networking and applications, pp 446–452
12. El-kenawy EST, El-Desoky AI, Al-rahamawy MF (2012) Extended max–min scheduling using petri net and load balancing. Int J Soft Comput Eng (IJSCE) 2(4):2231–2307

13. Banerjee S, Adhikari M, Kar S, Biswas U (2015) Development and analysis of a new cloudlet allocation strategy for QoS improvement in cloud. Arabian J Sci Eng 40(5):1409–1425. doi:10.1007/s13369-015-1626-9
14. Wang S, Liu Z, Sun Q, Zou H, Yang F (2012) Towards an accurate evaluation of quality of cloud service in service-oriented cloud computing. J Intell Manuf 25(2):283–291. doi:10.1007/s10845-012-0661-6
15. Rimal BP, Choi E, Lumb I (2010) A taxonomy, survey, and issues of cloud computing ecosystems. In: Antonopoulos N, Gillam L (eds) Cloud computing: principles systems and applications, Computer communications and networks. Springer, Berlin, pp 21–46. ISBN 978-1-84996-240-7. doi:10.1007/978-1-84996-241-4_2
16. Brucker P (2007) Scheduling algorithms, 5th edn. Springer, Berlin. ISBN 978-3-540-69515-8. doi:10.1007/978-3-540-69516-5
17. Contributed by techgreek.in (2010) Types of scheduling. 4th June
18. Mohan S (2009) Mixed scheduling, a new scheduling policy, proceedings of insight'09, 25–26 Nov 2009
19. Xiao Jun C, Jing Z, Junhuai L, Xiang L (2013) Resource virtualization methodology for on-demand allocation in cloud computing systems. SOCA 7:77–100. doi:10.1007/s11761-011-0092-9
20. Amalarethinam DIG, Muthulakshmi P (2011) An overview of the scheduling policies and algorithms in grid computing. Int J Res Rev Comput Sci 2(2):280–294
21. Khanli LM, Analoui M (2008) Resource scheduling in desktop grid by grid-JQA. In: The 3rd international conference on grid and pervasive computing workshops, GPC Workshops '08, Kunming, pp 63–68. doi:10.1109/GPC.WORKSHOPS.2008.27
22. Chatterjee T, Ojha VK, Adhikari M, Banerjee S, Biswas U, Snasel V (2014) Design and implementation of a new datacenter broker policy to improve the QoS of a cloud. In: © Springer international publishing Switzerland 2014, Proceedings of ICBIA 2014, advances in intelligent systems and computing vol 303, pp 281–290. doi:10.1007/978-3-319-08156-4_28
23. Belalem G, Tayeb FZ, Zaoui W (2010) Approaches to improve the resources management in the simulator CloudSim, First international conference, ICICA 2010, Tangshan, China, October 15–18, 2010 proceedings, LNCS, vol 6377, pp 189–196. doi:10.1007/978-3-642-16167-4_25
24. Calheiros RN, Ranjan R, De Rose CAF, Buyya R (2009) CloudSim: a novel framework formodeling and simulation of cloud computing infrastructures and services. In: Technical report, GRIDS-TR-2009-1. The University of Melbourne, Australia, Grid Computing and Distributed Systems Laboratory
25. Parsa S, Entezari-Maleki R (2009) RASA: a new grid task scheduling algorithm. Int J Digit Content Technol Appl 3:91–99
26. Yang J, Khokhar A, Sheikht S, Ghafoor A (1994) Estimating execution time for parallel tasks in heterogeneous processing (HP) environment. In: Proceedings of the heterogeneous computing workshop, Cancun, pp 23–28. doi:10.1109/HCW.1994.324966
27. Barham P, Dragovic B, Fraser K, Hand S, Harris T, Ho A, Neugebauer R, Pratt I, Warfield A (2003) Xen and the art of virtualization. In: Proceedings of the 19th ACM symposium on operating systems principles (SOS'2003). Bolton Landing, pp 177. ISBN:1-58113-757-5