Adaptive Cloudlet Scheduling Algorithm Using Three Phase Optimization Technique

Mohan Lavanya[1], B. Santhi[1], and Sivasankaran Saravanan[2](✉)

[1] School of Computing, SASTRA Deemed University, Thanjavur, Tamilnadu, India
m_lavanyass@ict.sastra.edu, shanthi@cse.sastra.edu

[2] College of Engineering, Debre Berhan University, Debre Berhan, Ethiopia
saran@dbu.edu.et

Abstract. The purpose of cloud computing is to give suitable access to the remote scattered resources. This is achieved through virtualization, which separates the physical computing resources into multiple virtual resources. The other technologies like grid, utility and distributed computing are the backbone of cloud computing. The scheduler plays important role because the user has to pay for the resource based on the time consumed during their usage. Currently, cloudlets and the virtual machines are scheduled according to FCFS and round robin which has higher latency. In order to reduce the latency and to have uniform distribution in scheduling the cloudlets to the Virtual Machines, this paper introduces called ACS3O algorithm which consists of 3 phases of optimization techniques using gang and dedicated processor scheduling to schedule the cloudlets. The proposed cloudlet scheduling algorithm optimizes few basic parameters like waiting time and makespan which have significant impact in the performance. Simulation is done in a Cloudsim environment to evaluate the proposed algorithm.

Keywords: Scheduling · Virtual machine · Adaptive approach · Cloudsim

1 Introduction

Cloud computing is an On-Demand service that provides infrastructure, software, platform as services, which is evolved from grid, utility and distributed computing. The cloud computing gives access to the remote scattered resources, which is achieved by Virtualization through virtual machine [4]. The Virtual Machine scheduler implements a suitable scheduling algorithm for executing various tasks [6] in an effective manner. Cloudlet Scheduler plays a crucial role because user has to pay for the resource based on the time consumed during their usage. In the existing cloudlet scheduling algorithm [5], there are various challenges in optimizing the parameters like waiting time, turnaround time, latency, and cost and load distribution. This proposed algorithm optimizes the parameters like waiting time and load distribution, resulting in reduced latency and cost in an effective manner.

In our paper, we proposed the Adaptive Cloudlet Scheduling with 3 Phase Optimization algorithm. The algorithm schedules the taskset based on categorization of the virtual machines and allocates the tasks to the VM with threshold value. We have calculated the waiting time and turnaround time. We observed the results of various dataset simulations and concluded that the proposed algorithm gives better result. Existing algorithm called RB2B performance compared with the proposed algorithm to shows the advantages of proposed method.

2 Existing System

Scheduling the cloudlet is mainly concerned in distributing the cloudlets to all the available VM, so that the cloud service is provided to user in a faster and effective manner. Thus, the scheduling plays a major role in cloud computing. Here a few existing scheduling algorithms are taken into account to analyze their performances and to compare with the proposed algorithm.

- FCFS
  This a static scheduling algorithm [1], in which the cloudlets are collected and queued until the VM are available and once the VM are free, cloudlet is assigned based on the arrival time. This is a simple scheduling algorithm. Certain smaller cloudlet waits for longer intervals.
- MIN-MIN
  This is a heuristic based scheduling algorithm [8] in which the smallest cloudlet is selected and assigned to a VM which gives the minimum completion time. In this, the longest cloudlet waits until the smaller cloudlets are scheduled.
- MAX-MIN
  It is also a heuristic based scheduling algorithm [2] in which the longest cloudlet is scheduled to the VM based on the minimum completion time. Here the smaller cloudlets starve, anyways it has a better makespan and throughput than the min-min algorithm.
- THROTTLED LOAD BALANCING ALGORITHM
  In this algorithm [7], an index table of VM and their states are maintained whenever a cloudlet arrives as it scans the index table to find a suitable VM. In this algorithm, every VM maintains a separate queue. Because of the queuing and scanning, the index table results in the increase in waiting time.
- SEQUENCE OPTIMISATION
  Here the cloudlets are scheduled, based on their arrival time in sequences. When a cloudlet arrives, it is allocated to be available VM in sequence. In each VM, the in and out parameters are calculated which are the arrival and burst time of that cloudlet. If the VM is busy, then the arriving cloudlet waits until the earlier cloudlet is scheduled, here time parameter is considered as waiting time of that cloudlet. In this, larger cloudlet may be allocated to lower MIPS VM and vice versa [3].

- RANGE WISE BUSY 2-WAY BALANCED ALGORTIHM
  In this algorithm [9], the VM are categorized based on the incoming cloudlet size. Whenever a cloudlet arrives, it checks whether a target VM is free, then the cloudlet is allocated to the target VM. Otherwise, VM is selected based on minimum finish time for that cloudlet. When the chosen VM is busy, the cloudlet is queued in the local queue. In case if the local queue gets filled, the cloudlet is queued to the global queue. Here the VM with higher MIPS gets overloaded with the cloudlets, while the VM with lower MIPS remains idle, resulting in poor resource utilization and waiting time is also increased.
- SLA-MCT ALGORITHM
  In this algorithm [10], single phase service level agreement-based algorithm proposed by authors. Whenever the cloudlet enters from broker system it will be taken for scheduling based on the agreement. If client request for maximum efficiency the tasks allocated to high speed machine. If the budget is mentioned the system which gives less rental are allocated. After allocation of the cloudlet to VM is over, the load of current load is added with the current execution time of VM. In paper [11] proposes the SLA-MCT algorithm for multi-cloud environment.

3 Proposed Architecture

The VM are categorized based on the cloudlet size and their load values are calculated. When a cloudlet arrives, it checks whether the target VM is free, if it is free cloudlet is allocated to that VM, otherwise it checks the load value. If the load value is minimum then it follows dedicated scheduling algorithm, otherwise it follows the gang scheduling algorithm to schedule the cloudlets. The VM is chosen based on the earliest finish time. If the chosen VM has maximum load value then the cloudlet is not allocated to that VM. It has the same time if chosen VM has the load value less than maximum threshold, its load value is calculated for that cloudlet. If the load value of chosen VM reaches the maximum load threshold then the cloudlet is not allocated to the chosen VM and it waits for the target VM. Otherwise, the cloudlet is allocated to the chosen VM. The proposed algorithm consists of three phases namely, (i) VM Categorization (ii) load calculation and (iii) adaptive scheduling. Based on the load values an adaptive approach is implemented in scheduling. The workflow of adaptive scheduling is described as follows: The VM are initialized, allotted to the hosts and are arranged in ascending order of processing speed. The incoming cloudlets from the global queue are sent to the datacentre broker. In the datacentre broker, the proposed scheduling algorithm is implemented. Initially the datacentre broker measures the cloudlet length and accordingly chooses a target VM for that cloudlet. If the target VM is available then the cloudlet is allocated to the VM, Otherwise the load value of VM is checked to determine which type of scheduling should be followed.

VM Categorization Phase

Here the VM are categorized based on the cloudlet acceptability range value. The initial stage involves in choosing a target VM for the arriving cloudlet based on their length. Datacentres broker defines the cloudlet acceptance range for each VM. The range value is obtained through a formula:

d = (cmax-cmin)/total mips
cmax = size of the largest cloudlet
cmin = size of the smallest cloudlet
total mips = $\sum_0^n speed\ of\ vm$i $0 < \mathrm{i} < = \mathrm{n}$

The cloudlet acceptance range for each VM consists of two values: one is lower limit and another one is upper limit.

Lower limit = cmin + S0d + S1d + ……. + Sm-1d + 1.
Upper Limit = cmin + S0d + S1d + ……. + Smd.

The datacentre broker matches the incoming cloudlet to its targeted VM based on their length.

Load Calculation Phase

This phase is crucial in this algorithm. The cloudlets are distributed to the various VM based on the calculated load values of the VM. This ensures that the VM with a higher speed (MIPS) is not overloaded. Hence their guaranteed utilization of resources properly. This phase involves,

(a) Calculate the load values for all the VM based on the cloudlets targeted at them.
(b) The datacentre broker checks whether the target VM is busy or not. If the target VM is not busy then the cloudlet is allocated to it, otherwise it checks the load value of the targeted VM. After the load calculation of all the VM, we determine the maximum load values as a maximum threshold and minimum load value as minimum threshold.

Adaptive Scheduling Phase

In adaptive scheduling the load value determines the type of scheduling that has to be implemented. If the load value is equal to minimum threshold then dedicated scheduling algorithm is used otherwise gang scheduling is followed. The workflow of the algorithm is as follows: If the targeted VM is busy then the load value is checked equal to minimum threshold, if it is so then the corresponding cloudlet is queued in the targeted VM. Otherwise the next best VM is searched based on the earliest finish time for that cloudlet. As a result, we find a chosen VM for that cloudlet. For the chosen VM the load value is again calculated for that cloudlet. If the calculated load value exceeds the maximum threshold then the cloudlet is queued in the target VM. The calculated load value of a VM is less than the maximum threshold then the cloudlet is queued in the chosen VM.

Flow chart for proposed method

Various conditions and its related paraments were shown in Fig. 1.

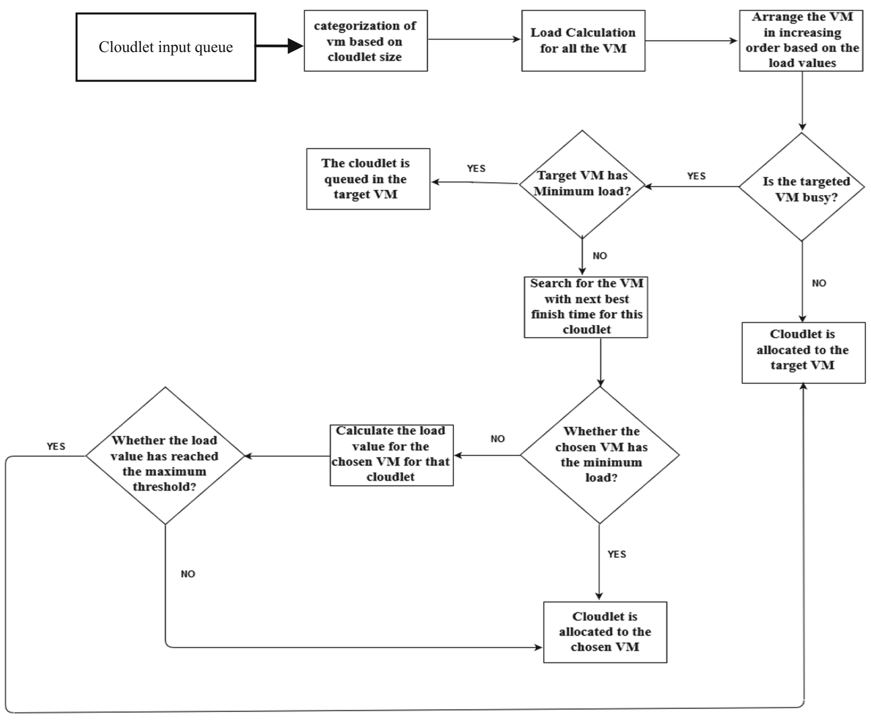

Cloudlet input queue
categorization of vm based on cloudlet size
Load Calculation for all the VM
Arrange the VM in increasing order based on the load values
Is the targeted VM busy?
YES
NO
Target VM has Minimum load?
YES
The cloudlet is queued in the target VM
NO
Search for the VM with next best finish time for this cloudlet
Cloudlet is allocated to the target VM
Whether the chosen VM has the minimum load?
NO
Calculate the load value for the chosen VM for that cloudlet
Whether the load value has reached the maximum threshold?
YES
NO
YES
Cloudlet is allocated to the chosen VM

Fig. 1. Flow chart of proposed method

Algorithm steps for proposed method

The proposed algorithm consists of N cloudlets and M number of VM.

INPUT: Cloudlet set (C_0,C_1….,C_n), N number of cloudlets, VM(VM_0,…..,VM_m), Speed of the VM, size of the cloudlets and their arrival time.

```
1. Arrange the VM in increasing order of their speeds.
2. Calculate the total speed (MIPS).
3. Find the Maximum Size Cloudlet.
4. Find the Minimum Size Cloudlet.
5. Calculate the Interval, d = (cmax-cmin)/total mips.
6. Calculate the upper and lower limit acceptance range values for all VM.
7. Find the target VM for each cloudlet
8. Calculate the Load Values for all the VM
9. Find the Maximum and Minimum threshold load valued VM.
10.  For all the cloudlets,
   10.1 If Target VM is free
   10.1.1  Cloudlet is allocated to that VM
   10.2  Else
      10.2.1  If Load value of target VM==min threshold load  then,
               The cloudlet is queued in the target VM
      10.2.2  Else
              Next best VM is chosen based on earliest finish time for that cloudlet.
            10.2.2.1 If chosen VM has minimum threshold then
                     The cloudlet is queued in the chosen VM
            10.2.2.2 Else
                     Calculate the load for the chosen VM
                     If calc load value>=max threshold then
                     Cloudlet is queued in the target VM
                     Else
                     Cloudlet is queued in the chosen VM.
```

4 Experimental Result

The proposed adaptive scheduling algorithm is simulated in cloudsim and a small dataset has been taken into account for illustration as shown in Table 1. Due to space constraints, we consider only 10 cloudlets and 3 VM. The cloudlets and VM are considered with the minimum length and their processing speeds. Cloudsim configuration is given in Table 2.

Table 1. Reference VM and cloudlets

Arrival time	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9
	0	1	1	2	2	3	5	6	8	9
Size (MI)	100	10	50	30	90	20	20	40	80	10
	VM0	VM1	VM2							
Processing speed (MIPS)	1	2	3							

Table 2. Cloudsim configuration

Elements	Parameters
Virtual machines	Image size = 1000 (MB), (VM memory) RAM = 512, Speed = x (MIPS), Bandwidth = 1000, Number of CPU's = 1, VMM name = "Xen"

Host	No of hosts and their host ID (Host memory)
	RAM = 2048 (MB),
	Storage = 1000000, Bandwidth = 10000

Cloudlets	Length = 1000 (MI), Input and Output File Size, No. of processing elements = 1
Data centre	System architecture = "x86" Operating system = "Linux" VMM = "Xen", Time zone = 10.0 Cost Per Memory = 0.05, Cost Per Storage = 0.001

VM Categorization:
cmax = maximum size cloudlet = 100
cmin = minimum size cloudlet = 10
 total mips = (1 + 2 + 3) = 6
 d = (cmax-cmin)/total mips = (100 − 10)/6 = 15
Cloudlet acceptance range
VM0 lower limit = 10, upper limit = 25(10 + d)
VM1 lower limit = 25 + 1 = 26, upper limit = 55(25 + 2d)
VM2 lower limit = 55 + 1 = 56, upper limit = 100(55 + 3d)

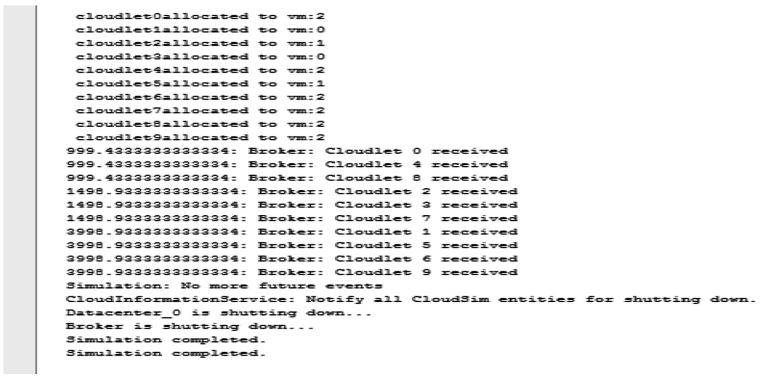
Load Calculation:

VM0 = C1, C5, C6, C9
VM1 = C2, C3, C7
VM2 = C0, C4, C8
Calculation of load values for each VM
 VM0 = (10 + 20 + 20 + 10)/1 = 60
 VM1 = (50 + 30 + 40)/2 = 60
 VM2 = (100 + 90 + 80)/3 = 89.9
 Max threshold load = 89.9 and Min threshold load = 60

Adaptive Scheduling:

Co arrives → Allocated to VM2 (sinceVM2 is free)
C1 arrives → Allocated to VM0 (sinceVM0 is free)
C2 arrives → Allocated to VM1 (sinceVM1 is free)
C3 arrives → Actual targeted VM is VM1 (since VM1 is busy)
Check Load value of VM1 = = min threshold load
Follow dedicated scheduling (i.e.) C3 is queued to VM1
C4 arrives → Actual targeted VM is VM2 (since VM2 is busy)
 Check load value of VM2 = = max threshold load
 Calculate the expected earlier finish time for C4 in all VM

VM0 - (10 + 90)/1 = 100, VM1 - (80 + 90)/2 = 85, VM2 - (100 + 90)/3 = 63.33
So, C4 is queued to VM2.
C5 arrives → Actual targeted VM is VM0 (since VM0 is busy)
Check Load value of VM0 = = min threshold load



```
 cloudlet0allocated to vm:2
 cloudlet1allocated to vm:0
 cloudlet2allocated to vm:1
 cloudlet3allocated to vm:0
 cloudlet4allocated to vm:2
 cloudlet5allocated to vm:1
 cloudlet6allocated to vm:2
 cloudlet7allocated to vm:2
 cloudlet8allocated to vm:2
 cloudlet9allocated to vm:2
999.4333333333334: Broker: Cloudlet 0 received
999.4333333333334: Broker: Cloudlet 4 received
999.4333333333334: Broker: Cloudlet 8 received
1498.9333333333334: Broker: Cloudlet 2 received
1498.9333333333334: Broker: Cloudlet 3 received
1498.9333333333334: Broker: Cloudlet 7 received
3998.9333333333334: Broker: Cloudlet 1 received
3998.9333333333334: Broker: Cloudlet 5 received
3998.9333333333334: Broker: Cloudlet 6 received
3998.9333333333334: Broker: Cloudlet 9 received
Simulation: No more future events
CloudInformationService: Notify all CloudSim entities for shutting down.
Datacenter_0 is shutting down...
Broker is shutting down...
Simulation completed.
Simulation completed.
```

Fig. 2. Cloudlet output results

Table 3. Experimental results

Algorithms	Average wait time	Total execution time
In paper [1]	15.34	90
Adaptive (proposed model)	14.34	89.9

Follow dedicated scheduling (i.e.) C5 is queued to VM0
C6 arrives → Actual targeted VM is VM0 (since VM0 is busy)
Check Load value of VM0 = = min threshold load
Follow dedicated scheduling (i.e.) C6 is queued to VM0
C7 arrives → Actual targeted VM is VM1 (since VM1 is busy)
Check load value of VM1 = = min threshold load
Follow dedicated scheduling (i.e.) so, C7 is queued to VM1
C8 arrives → Actual targeted VM is VM2 (since VM2 is busy)
Check load value of VM2 = = max threshold load
Calculate the expected earlier finish time for C8 in all VM
VM0 - 30 + 80 = 110, VM1 - (70 + 40) = 110, VM2 - (63.3 + 26.6)/3 = 89.9
So, C8 is queued to VM2.
C9 arrives → Actual targeted VM is VM0 (since VM0 is busy)
Check Load value of VM0 = = min threshold load,
C9 is queued to VM0.

Figure 2 shows cloudlet output results for the given data. Table 3 illustrates the inference made is that the proposed adaptive scheduling algorithm has minimum average waiting time than that of existing [1] algorithm. The cloudlets are uniformly distributed and the VM with the higher MIPS is not overloaded, hence load balancing is achieved.

5 Conclusion

In the above proposed Adaptive scheduling algorithm, the cloudlets are distributed to all the VM based on their load values. From the deep experimental analysis and comparative survey, the proposed algorithm proves it has less average waiting time and resources are properly utilized. Here all the VM are utilized uniformly irrespective of their speed, so VM with higher MIPS are never overloaded at the same time VM with lower MIPS does not remains idle. Thus, this provides better performance as the VM are properly utilized. This work can be extended to dynamically schedule cloudlets and updating the load values of the VM by learning techniques.

References

1. Roy, S., Banerjee, S., Chowdhury, K.R., Biswas, U.: Development and analysis of a three-phase cloudlet allocation algorithm. J. King Saud Univ. Comput. Inf. Sci. **29**(4), 473–483 (2017)
2. Lavanya, M., Sahana, V., Swathi Rekha, K., Vaithiyanathan, V.: Adaptive load balancing algorithm using modified resource allocation strategies on infrastructure as a service cloud systems. ARPN J. Eng. Appl. Sci. **10**(10), 4522–4526 (2015)
3. Agarwal, A., Jain, S.: Efficient optimal algorithm of task scheduling in cloud computing environment. Int. J. Comput. Trends Technol. (IJCTT) **9**(7), 344–349 (2014)
4. Bhavani, B.H., Guruprasad, H.S.: Resource provisioning techniques in cloud computing environment: a survey. Int. J. Res. Comput. Commun. Technol. **3**(3), 395–401 (2014)
5. Chawla, Y., Bhonsle, M.: A study on scheduling methods in cloud computing. Int. J. Emerg. Trends Technol. Comput. Sci. **1**(3), 12–17 (2012)
6. Mathew, T., Sekaran, K.C., Jose, J.: Study and analysis of various task scheduling algorithms in the cloud computing environment. In: International Conference on Advances in Computing, Communications and Informatics (ICACCI) (2014)
7. Tohidirad, Y., Abdezadeh, S., Soltani Aliabadi, Z., Azizi, A., Moradi, M.: Virtual machine scheduling in cloud computing environment. Int. J. Manag. Public Sect. Inf. Commun. Technol. **6**(4), 1–6 (2015)
8. Liu, G., Li, J., Xu, J.: An improved min-min algorithm in cloud computing. In: Du, Z. (ed.) Proceedings of the 2012 International Conference of Modern Computer Science and Applications. AISC, vol. 191, pp. 47–52. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-33030-8_8

9. Mao, Y., Chen, X., Li, X.: Max–min task scheduling algorithm for load balance in cloud computing. In: Patnaik, S., Li, X. (eds.) Proceedings of International Conference on Computer Science and Information Technology. AISC, vol. 255, pp. 457–465. Springer, New Delhi (2014). https://doi.org/10.1007/978-81-322-1759-6_53
10. Domanai, S.G., Reddy, G.R.M.: Load balancing in cloud computing using modified throttled algorithm. In: IEEE International Conference on Cloud Computing in Emerging Markets (CCEM) (2013)
11. Teyeb, H.: Integrated optimization in cloud environment, Networking and Internet architecture. Université Paris-saclay (2017)