"A COMPARATIVE ANALYSIS OF LOAD BALANCING ALGORITHMS WITH CLOUDSIM"

[1]BHUSHAN LANDE
Department of Information Technology, Shri Sant Gajanan Maharaj College of Engineering, Shegaon Maharashtra, India
bhushanlande231198@gmail.com,

[2]TOUKIR DOSANI
Department of Information Technology, Shri Sant Gajanan Maharaj College of Engineering, Shegaon Maharashtra, India
taukir2999@gmail.com,

[3]DHIRAJ KHARODE
Department of Information Technology, Shri Sant Gajanan Maharaj College of Engineering, Shegaon Maharashtra, India
kharodedhiraj100@gmail.com,

[4]JAGRUTI KALAMBE
Department of Information Technology, Shri Sant Gajanan Maharaj College of Engineering, Shegaon Maharashtra, India
jagrutihkalambe2015@gmail.com and

[5]MOHIT AGRAWAL
Department of Information Technology, Shri Sant Gajanan Maharaj College of Engineering, Shegaon Maharashtra, India
mohita2266@gmail.com

[6]PROF. PRITI KALE
Department of Information Technology, Shri Sant Gajanan Maharaj College of Engineering, Shegaon Maharashtra, India
pritikale@gmail.com

ABSTRACT: *Cloud load balancing helps to manage workload demands by distributing resources among various computers or servers in a cloud computing environment. There are many challenges in terms of performance and efficiency in the cloud environment. The aim of the paper is to address the issue of distribution of cloudlets over virtual machine with maximum efficiency and throughput. The prime goal is to address the load distribution in multiple Virtual Machines and to propose an algorithm which has minimum response time and minimum power consumption using CloudSim.*

Keywords: Cloud Computing, load balancing, Virtual Machines (VM's), CloudSim

1. INDRODUCTION

Cloud Computing is an emerging technology in IT environment having large requirement of resource and infrastructure. Load balancing in the cloud environment can be defined as the process of distributing or sharing the workload amongst various computer resources (servers, computers etc.). In respect to cloud computing, load balancing is a very vital task.

With an efficient load balancing techniques, the throughput can be increased and response time can be minimized. With the increase in number of users on cloud, there is a decrease in the number of resources that gives rise to the time delay in providing service to the users. In such a situation there may arise the problem of underload and overload which can be dealt with the efficient load balancing algorithm. A load balancing algorithm which is dynamic in nature does not consider the previous state of the system, that is it considers the present state of the system.

In this paper, a comparative analysis between various load balancing algorithms like First Come First Serve (FCFS), General Prioritized Load balancing algorithm and A Priority Based Dynamic Resource Mapping Algorithm is presented. The performance of the mentioned algorithm is studied and checked through the overall execution time taken by each algorithm to complete the task using CloudSim (version 3.0)

2. LITERATURE REVIEW

Authors described about an improved algorithm of dynamic resource allocation [1] in cloud computing considering tasks' priority and balancing load by sorting the virtual machines on the basis of the processing power, job requests on the basis of the number of instructions and priority and then assigning group of cloudlets to the corresponding virtual machine.

The proposed algorithm et al. [2] proposed an approach for Dynamic Load Balancing in Cloud Computing uses agents. It has been observed that performance of the existing load balancing algorithms can be improved with the help of mobile agents. Mobile agents can be defined as composition of data and software which can drift from one machine to another. The total task is completed using agents i.e Regular and Mobile. Mobile agents can be defined as a constitution of Computer data and software which can drift from one computer to another. The comparison is done between the traditional scheme used for load balancing and the proposed algorithm using mobile agents and it has been observed that the proposed protocol approach has yield greater results by not only just reducing the communication cost of server but also by accelerating the rate of load balancing. This results in the improvement of the Response Time and Throughput of the cloud.

An approach to balance the work load by arranging virtual machine on the basis of their processing power and arranging the cloudlets which is then submitted to broker for allocation using genetic algorithm in[3].

A new priority based job scheduling algorithm [4] is proposed which is based on multiple criteria decision making model. It describes that particular job scheduling algorithm in cloud environment should pay attention to multi attribute and multi criteria properties of jobs.

3. OTHER DETAILS

CLOUDSIM 3.0: CloudSim is a simulation tool that enables the modeling of allocation policies in a repetitive and under controlled environment without any cost [5]. It is a simulator; hence it doesn't run any actual software .CloudSim is a library for simulation of cloud scenarios. It consists of important classes which can be used to describe data centers, computational resources, Virtual Machines, applications, users along with the policies for management of different parts of the system such as resource allocation policies. It also supports the creation of Datacenter, DataCenter brokers, etc. Either NetBeans or Eclipse IDE can be used as they both support java language. In this paper we implemented CloudSim using Eclipse IDE.

3.1. CLOUDSIM TERMINOLOGY'S AND CONFIGURATION

Cloudlets: In CloudSim, task is represented by cloudlets.

Users: The task is sent to the cloud through the users.
Datacenter: It is just the repository of the virtual servers that host various applications.

Broker: Broker acts as an intermediate used to handle the incoming requests and then forward it to the host.

Virtual Machines (VM's): Virtual machines are the virtual computers that provide the functionality like physical computers.

They perform the execution of the cloudlets and also shows resource consumption. [6]

Host: It is used to manage the virtual machine (e.g. creation and destruction)

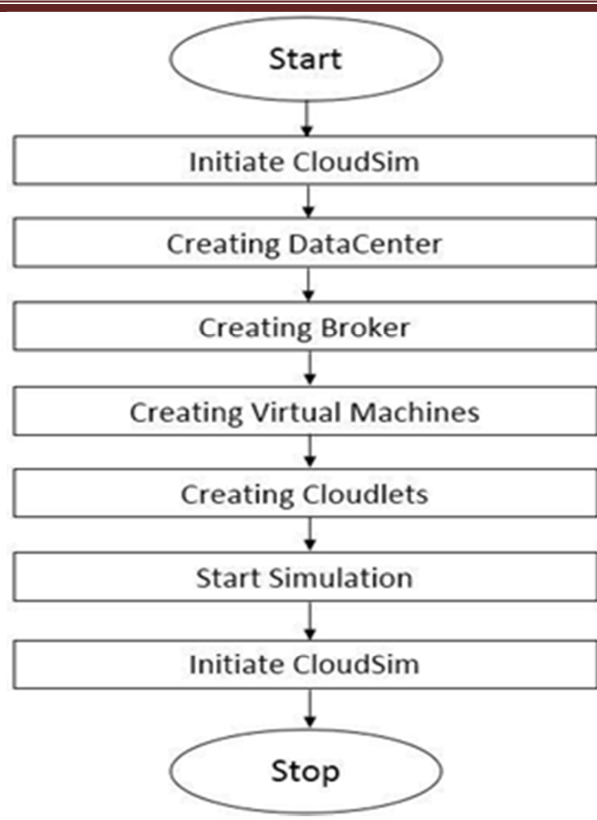


Figure 1: CloudSim Lifecycle

3.2. CLOUDSIM CONFIGURATION: The following table gives a description of the configuration setup of CloudSim which includes the no. of cloudlets distributed among different virtual machines as well as the no. of users. It also discusses about the configuration of the system to create a datacenter.

Table 1: Simulation Parameters

PARAMETERS	VALUES
Number of Users	1
Number of VM's	5
Number of Cloudlets	50

Table 2: VM Details

PARAMETERS	VALUES
Number of VM's	5
Name of VM	VM0, VM1, VM2 , VM3 and VM4
RAM	512
MIPS	451, 250, 651, 50, 851.
BW	1000

Table 3: Host Details

PARAMETERS	VALUES
Number of Datacenters	1
RAM	102400
Storage	1000000
BW	200000

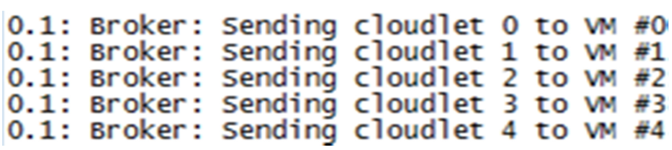
4. SIMULATION OF VARIOUS LOAD BALANCING ALGORITHMS

We have tried to analyze different load balancing algorithms such as Dynamic priority, FCFS and general priority in this section.
The results obtained after simulating three different algorithms in the CloudSim Simulator is as follows:

FCFS (First Come First Serve)

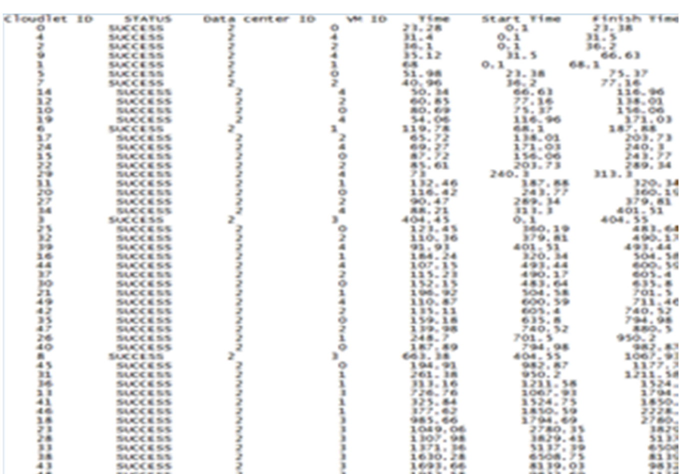
First come first serve is used for the resource allocation so that the waiting queue time is least and can be selected for the next upcoming task. Allocation of application-specific tasks to host the cloud based data center is the responsibility of virtual machine provisioned component.

This algorithm was put under simulation and following output was observed: -



```
0.1: Broker: Sending cloudlet 0 to VM #0
0.1: Broker: Sending cloudlet 1 to VM #1
0.1: Broker: Sending cloudlet 2 to VM #2
0.1: Broker: Sending cloudlet 3 to VM #3
0.1: Broker: Sending cloudlet 4 to VM #4
```

Figure 2: Allocation of cloudlet in FCFS



Cloudlet ID	STATUS	Data center ID	VM ID	Time	Start Time	Finish Time
0	SUCCESS	2	0	23.28	0.1	23.38
4	SUCCESS	2	4	31.4	0.1	31.5
2	SUCCESS	2	2	36.1	0.1	36.2
9	SUCCESS	2	4	35.12	31.5	66.63
1	SUCCESS	2	1	68	0.1	68.1
5	SUCCESS	2	0	51.98	23.38	75.37
7	SUCCESS	2	2	40.96	36.2	77.16
14	SUCCESS	2	4	50.34	66.63	116.96
12	SUCCESS	2	2	60.85	77.16	138.01
10	SUCCESS	2	0	80.69	75.37	156.06
19	SUCCESS	2	4	54.06	116.96	171.03
6	SUCCESS	2	1	119.78	68.1	187.88
17	SUCCESS	2	2	65.72	138.01	203.73
24	SUCCESS	2	4	69.27	171.03	240.3
15	SUCCESS	2	0	87.72	156.06	243.77
22	SUCCESS	2	2	85.61	203.73	289.34
29	SUCCESS	2	4	73	240.3	313.3
11	SUCCESS	2	1	132.46	187.88	320.34
20	SUCCESS	2	0	116.42	243.77	360.19
27	SUCCESS	2	2	90.47	289.34	379.81
34	SUCCESS	2	4	88.21	313.3	401.51
3	SUCCESS	2	3	404.45	0.1	404.55
25	SUCCESS	2	0	123.45	360.19	483.64
32	SUCCESS	2	2	110.36	379.81	490.17
39	SUCCESS	2	4	91.93	401.51	493.44
16	SUCCESS	2	1	184.24	320.34	504.58
44	SUCCESS	2	4	107.15	493.44	600.59
37	SUCCESS	2	2	115.23	490.17	605.4
30	SUCCESS	2	0	152.15	483.64	635.8
21	SUCCESS	2	1	196.92	504.58	701.5
49	SUCCESS	2	4	110.87	600.59	711.46
42	SUCCESS	2	2	135.11	605.4	740.52
35	SUCCESS	2	0	159.18	635.8	794.98
47	SUCCESS	2	2	139.98	740.52	880.5
26	SUCCESS	2	1	248.7	701.5	950.2
40	SUCCESS	2	0	187.89	794.98	982.87
8	SUCCESS	2	3	663.38	404.55	1067.93
45	SUCCESS	2	0	194.93	982.87	1177.[illegible]
31	SUCCESS	2	1	261.38	950.2	1211.58
36	SUCCESS	2	1	313.16	1211.58	1524.[illegible]
13	SUCCESS	2	3	726.76	1067.93	1794.[illegible]
41	SUCCESS	2	1	325.84	1524.75	1850.[illegible]
46	SUCCESS	2	1	377.62	1850.59	2228.[illegible]
18	SUCCESS	2	3	985.66	1794.69	2780.[illegible]
23	SUCCESS	2	3	1049.06	2780.35	3829[illegible]
28	SUCCESS	2	3	1307.98	3829.41	5137[illegible]
33	SUCCESS	2	3	1371.36	5137.39	6508[illegible]
38	SUCCESS	2	3	1630.28	6508.75	8139[illegible]
43	SUCCESS	2	3	1693.66	8139.03	9832[illegible]
48	SUCCESS	2	3	1952.58	9832.69	1178[illegible]

Figure 3: Total Execution Time of FCFS

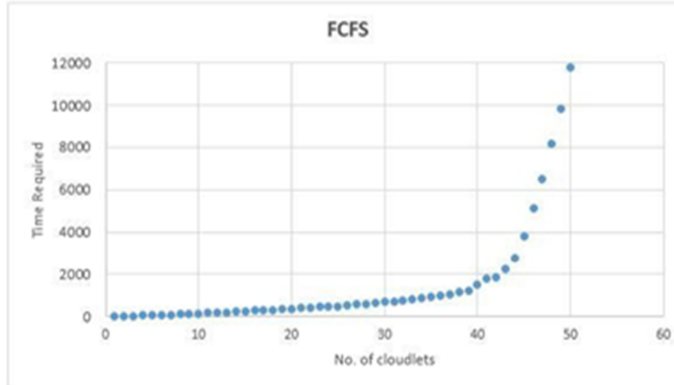
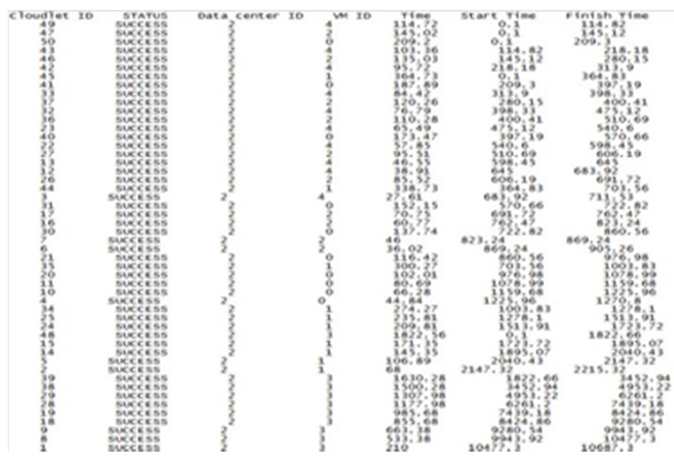


Figure 4: Graph of Time required to process in FCFS

GP (Generalized Priority)

This is a priority-based algorithm in which the tasks are allocated to the VM on the basis of the priority of the cloudlet. Following were the output observed: -



Cloudlet ID	STATUS	Data center ID	VM ID	Time	Start Time	Finish Time
49	SUCCESS	2	4	114.72	0.1	114.82
[illegible]	SUCCESS	2	2	145.02	0.1	145.12
50	SUCCESS	2	[illegible]	209.1	0.1	209.2
43	SUCCESS	2	4	103.36	114.82	218.18
42	SUCCESS	2	2	135.03	145.12	280.15
45	SUCCESS	2	4	95.72	218.18	313.9
[illegible]	SUCCESS	2	0	364.73	0.1	364.83
[illegible]	SUCCESS	2	[illegible]	187.69	209.3	397.19
33	SUCCESS	2	4	84.42	313.9	398.33
32	SUCCESS	2	2	120.26	280.15	400.41
37	SUCCESS	2	4	76.79	398.33	475.12
36	SUCCESS	2	2	110.28	400.41	510.69
[illegible]	SUCCESS	2	[illegible]	[illegible]	475.12	540.6
40	SUCCESS	2	[illegible]	173.47	397.19	570.66
[illegible]	SUCCESS	2	4	57.85	540.6	598.45
[illegible]	SUCCESS	2	2	95.5	510.69	606.19
[illegible]	SUCCESS	2	4	46.55	598.45	645
[illegible]	SUCCESS	2	[illegible]	38.91	645	683.92
[illegible]	SUCCESS	2	[illegible]	85.53	606.19	691.72
[illegible]	SUCCESS	2	[illegible]	338.73	364.83	703.56
44	SUCCESS	2	4	27.61	683.92	711.53
[illegible]	SUCCESS	2	[illegible]	152.15	570.66	722.82
[illegible]	SUCCESS	2	[illegible]	70.75	691.72	762.47
[illegible]	SUCCESS	2	[illegible]	[illegible]	703.56	823.24
[illegible]	SUCCESS	2	[illegible]	137.74	722.82	860.56
[illegible]	SUCCESS	2	2	46	823.24	869.24
[illegible]	SUCCESS	2	[illegible]	36.02	869.24	905.26
21	SUCCESS	2	0	116.42	860.56	976.98
35	SUCCESS	2	1	300.27	703.56	1003.83
20	SUCCESS	2	0	102.01	976.98	1078.99
11	SUCCESS	2	0	80.69	1078.99	1159.68
10	SUCCESS	2	0	66.28	1159.68	1225.96
4	SUCCESS	2	0	44.84	1225.96	1270.8
34	SUCCESS	2	1	274.27	1003.83	1278.1
25	SUCCESS	2	1	235.81	1278.1	1513.91
24	SUCCESS	2	1	209.81	1513.91	1723.72
48	SUCCESS	2	3	1822.56	0.1	1822.66
15	SUCCESS	2	1	171.35	1723.72	1895.07
14	SUCCESS	2	1	145.35	1895.07	2040.43
5	SUCCESS	2	1	106.89	2040.43	2147.32
2	SUCCESS	2	1	68	2147.32	2215.32
39	SUCCESS	2	3	1630.28	1822.66	3452.94
38	SUCCESS	2	3	1500.28	3452.94	4953.22
29	SUCCESS	2	3	1307.98	4953.22	6261.2
28	SUCCESS	2	3	1177.98	6261.2	7439.18
19	SUCCESS	2	3	985.68	7439.18	8424.86
18	SUCCESS	2	3	855.68	8424.86	9280.54
9	SUCCESS	2	3	663.38	9280.54	9943.92
8	SUCCESS	2	3	533.38	9943.92	10477.3
1	SUCCESS	2	3	210	10477.3	10687.3

Figure 5: Execution time taken by GP Algorithm

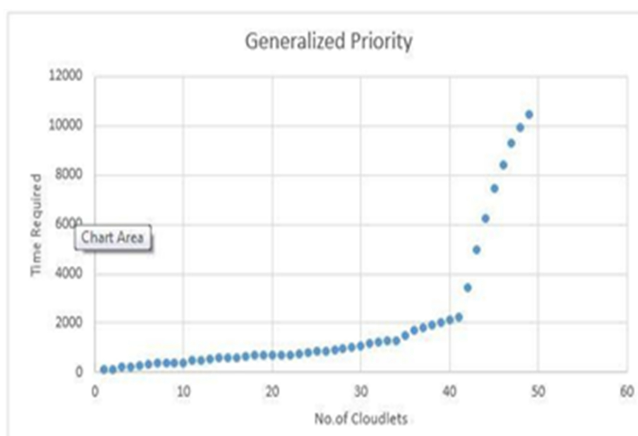


Figure 6: Graph of Time required to process in Priority

PBDRM (Priority Based Dynamic Resource Mapping Algorithm)

In dynamic priority based load balancing algorithm, the cloudlets are distributed among Virtual Machines according to their capacities. This is an improved algorithm in which resource allocation is done dynamically on the basis of the priority of the task and balancing the load which will maximize the throughput and minimize the execution time. It sorts the virtual machines and cloudlets according to their processing powers and no. of instruction and priority respectively. Tasks are allocated accordingly [1].The performance of the algorithm is judged on the basis of their execution time and compared with others. Following were the outputs observed:-

```
0.1: Broker: Sending cloudlet 3 to VM #3
0.1: Broker: Sending cloudlet 6 to VM #3
0.1: Broker: Sending cloudlet 4 to VM #3
0.1: Broker: Sending cloudlet 2 to VM #3
0.1: Broker: Sending cloudlet 1 to VM #3
0.1: Broker: Sending cloudlet 22 to VM #1
0.1: Broker: Sending cloudlet 17 to VM #1
0.1: Broker: Sending cloudlet 20 to VM #1
0.1: Broker: Sending cloudlet 15 to VM #1
0.1: Broker: Sending cloudlet 18 to VM #1
0.1: Broker: Sending cloudlet 13 to VM #1
0.1: Broker: Sending cloudlet 16 to VM #1
0.1: Broker: Sending cloudlet 11 to VM #1
0.1: Broker: Sending cloudlet 14 to VM #1
0.1: Broker: Sending cloudlet 9 to VM #1
0.1: Broker: Sending cloudlet 32 to VM #0
0.1: Broker: Sending cloudlet 27 to VM #0
0.1: Broker: Sending cloudlet 30 to VM #0
0.1: Broker: Sending cloudlet 25 to VM #0
0.1: Broker: Sending cloudlet 28 to VM #0
0.1: Broker: Sending cloudlet 23 to VM #0
0.1: Broker: Sending cloudlet 26 to VM #0
```

Figure 7: Allocation of cloudlets in PBDRM

Cloudlet ID	STATUS	Data center ID	VM ID	Time	Start Time
49	SUCCESS	2	4	114.72	0.1
42	SUCCESS	2	2	125.13	0.1
32	SUCCESS	2	0	144.89	0.1
22	SUCCESS	2	1	196.92	0.1
47	SUCCESS	2	4	110.93	114.82
37	SUCCESS	2	2	120.26	125.23
27	SUCCESS	2	0	137.86	144.99
50	SUCCESS	2	4	110.87	225.76
40	SUCCESS	2	2	120.18	245.49
17	SUCCESS	2	1	184.24	197.02
30	SUCCESS	2	0	137.74	282.85
45	SUCCESS	2	4	107.15	336.63
35	SUCCESS	2	2	115.31	365.67
48	SUCCESS	2	4	107.08	443.78
25	SUCCESS	2	0	130.72	420.59
20	SUCCESS	2	1	184.02	381.26
38	SUCCESS	2	2	115.23	480.98
43	SUCCESS	2	4	103.36	550.86
12	SUCCESS	2	3	662.28	0.1
28	SUCCESS	2	0	130.6	551.31
33	SUCCESS	2	2	110.36	596.21
15	SUCCESS	2	1	171.35	565.28
46	SUCCESS	2	4	103.3	654.22
23	SUCCESS	2	0	123.57	681.9
36	SUCCESS	2	2	110.28	706.57
41	SUCCESS	2	4	99.57	757.52
18	SUCCESS	2	1	171.13	736.63
31	SUCCESS	2	2	105.41	816.85
26	SUCCESS	2	0	123.45	805.47
44	SUCCESS	2	4	99.51	857.09
34	SUCCESS	2	2	105.33	922.25
21	SUCCESS	2	0	116.42	928.92
39	SUCCESS	2	4	95.79	956.6
13	SUCCESS	2	1	158.46	907.77
29	SUCCESS	2	2	100.46	1027.58
24	SUCCESS	2	0	116.3	1045.34
16	SUCCESS	2	1	158.24	1066.22
7	SUCCESS	2	3	598.91	662.38
19	SUCCESS	2	0	109.28	1161.64
11	SUCCESS	2	1	145.57	1224.47
14	SUCCESS	2	1	145.35	1370.04
9	SUCCESS	2	1	132.68	1515.39
10	SUCCESS	2	3	597.84	1261.29
5	SUCCESS	2	3	534.46	1859.12
8	SUCCESS	2	3	533.38	2393.58
3	SUCCESS	2	3	470	2926.96
6	SUCCESS	2	3	468.92	3396.96
4	SUCCESS	2	3	404.46	3865.88
2	SUCCESS	2	3	340	4270.34
1	SUCCESS	2	3	210	4610.34

Simulation Completed

Figure 8: Total Execution time of Priority based Dynamic Resource mapping Algorithm

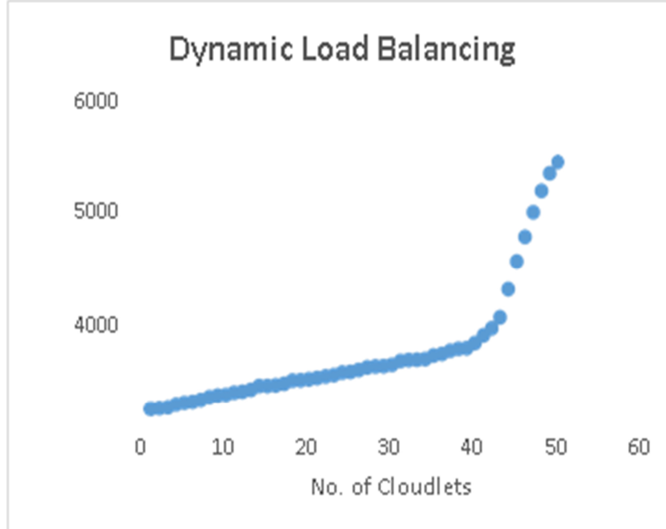


Figure 9: Graph of Time required to process in PBDRM

5. ANALYSIS AND SIMULATION RESULT

Table 4: Simulation Results

Algorithm	Total Execution Time(m/s)
FCFS	11785.27
GP	10687.3
PBDRM	4820.34

Execution Time= Start Time + Time (Length of Cloudlet).

Total Execution Time = Sum of Execution times of total cloudlets.

With the numerical figures we can clearly see that PBDRM is faster and took less time to complete all the cloudlet execution. FCFS took the most time to complete the execution. So, with the above result we can analyze that PBDRM is the most efficient algorithm.

6. CONCLUSION AND FUTURE WORK

In this paper we came across three load balancing algorithm which is First Come First Serve, General Priority and Priority Based Dynamic Resource Mapping. These algorithms were implemented using CloudSim Simulator and different outputs were observed. Afterwards, the results were observed and analyzed. In future, more efficient sorting algorithm can be proposed which will reduce the execution time and maximize the throughput and accordingly decrease the power consumption and response time.

6. REFERENCES

[1] Farzana Sadia1, Nusrat Jahan2, Lamisha Rawshan3, Madina Tul Jeba4 and Dr. Touhid Bhuiyan (2017): A Priority Based Dynamic Resource Mapping Algorithm For Load Balancing In Cloud

[2]Anjali, Grover J., Singh M., Singh C., and Sethi H. (2015): A new approach for dynamic load balancing in cloud computing, Proc. Nat. Conf. Adv. Eng. Technol. Manage

[3] Monika Lagwal(2017): A load balancing algorithm in cloud computing using genetic algorithm

[4] Shamsollah Ghandari, Mohamed Othman(2012): A priority based job scheduling algorithm in cloud computing

[5] R. N. Calheiros, Rajiv Ranjan, Anton Beloglazov, C.A.F. De Rose, Rajkumar Buyya, "CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms", Software Practice and Experience, Wiley publishers, 2010.

[6] Tarun goyal, Arjit Singh, Aakanksha Agrawal, Cloudsim:simulator for cloud computing infrastructure and modeling, ICMOC-2012.