TASK CLASSIFICATION USING MULTIQUEUE INTERLACING PEAK SCHEDULING METHOD IN CLOUD

Madhuri H. Patil
PG student, Department of Computer Engineering, SSVPS's B. S. Deore College of Engineering, Dhule

ABSTRACT

In cloud computing, resource requirement for tasks is vast and it may have ambiguous factors and changes dynamically over time. These factors causes load imbalances. A scheduling technique known as interlacing peak is developed during which first, the resource requirement of task is obtained using information such as CPU, I/O and memory information is gathered continuously after certain time period then all task are classified into three queues i.e CPU intensive, I/O intensive and memory intensive, then after that resources are being sorted in ascending order into again three queues CPU intensive, I/O intensive and memory intensive which is performed by global resource manager. After all this the task is being scheduled for its execution according to its intensity for CPU, I/O and memory usage. Intensity is determined on the basis of CPU capacity, I/O operations and memory usage. In another words, low CPU intensity tasks are scheduled with resources with low CPU utilization then it matched with resources of CPU intensive queue, low I/O intensity tasks are scheduled with resources with shorter I/O wait times are matched with resources with I/O intensive queue and low memory intensity tasks are scheduled with resources that have low memory usage are matched up with resources of memory intensive queue. The proposed system can balance loads especially true when resources are less often and many tasks will demand for the same resource.

Index Terms

Cloud computing, load balancing, multiqueue, task classification

1. INTRODUCTION

Cloud computing has wide variety of applications that can be used as an tasks those can demand for various types resources for its completion of its execution, some may require large amount of memory and some may need large amount of CPU utilization and others may require greater amount of I/O operations. This all things can load imbalancing [1]

However There are so many dynamic and ambiguous attributes related to task and resources such as uncertain changes in resource requirement by task. Some task may require resources at any time during its life time, those resources may available or unavailable to any task at any time. If there could be more resources than task, then much of resources could get wasted or it may happen that tasks are more than number of resources then scheduling performance
could get affected. This all factors got into load imbalancing and also resource utilization could get badly impacted [2].
Task scheduling algorithm is a method by which tasks are allocated to proper required task from data center. The task scheduling method improves average response time as rate of arrival of task could gets higher.

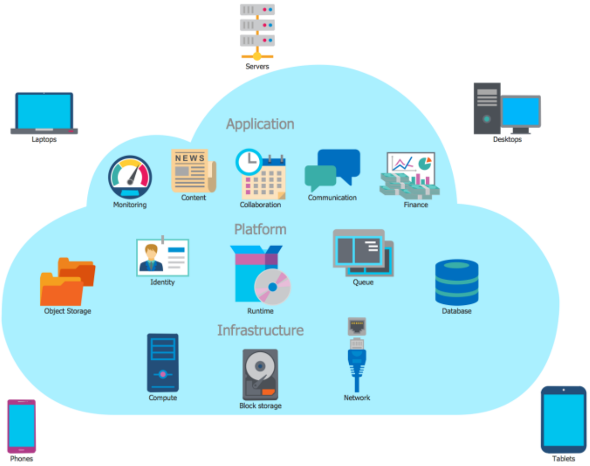

Figure 1: cloud services

2. RELATED WORK

There are three types of task scheduling based on scheduling methods in cloud computing. The first category consists of scheduling methods on the basis of time, including the response times, the best time span, and the completion time. The second is on the basis of performance, such as load balancing and resource utilization. The third is multiobjective optimization, which includes the budget cost, QoS, and energy consumption.

In 2011, Boutaba et al. developed a method to classify tasks & algorithm establishes dual fairness constraint in which first constraint is to classify user task by quality of services then general expectation function is established with classification of task to restrain the fairness of resources in selection method & second constraint is to outline resource fairness function to evaluate the fairness of resource allocation [3].

In 2011, Qi Zhang et al. focused on characterizing run-time task resources usage for CPU, memory & disk, to find an accurate characterization that can reproduce performance of workload traces in terms of key performance metrics such as task wait time & machine resource utilization [4].

In 2012, Xifeng yan et al proposed method for prediction of task characterization for efficiently provisioning computing resources in the cloud, there is need of capability of characterizing & predicting workload on virtual machine [5].

In 2013, Zhang et al. proposed method in which workload is divided into distinct task classes by using k- means clustering algorithm on the basis of similar characteristics of requirement for resources & performances [6].

In 2013, Moreno et al. analyzed task characteristics & established a model to simulate resource usage patterns & predict resources to optimize resource usage in which a approach for characterizing workloads that considers workloads in context of both user & task to capture resource estimation & utilization patterns [7].

In 2013, Malte Schwarzkopf proposes approach for task characterization for allocation, distinguishing between CPU & memory – intensified job, large & small jobs & so on , in this for addressing needs of parallelism, shared state & lock – free concurrency control [8].

3. SYSTEM MODEL

The system model describes the definitions of task and resources and queues of resources and tasks. The variables and their appropriate meaning mentioned in Table 1.

Table 1: Main Notation Definitions

System	Definitions
U_i	Resources I, 1≤i≤N
T_j	Task j, 1≤j≤K
N, K	Defines resources and tasks respectively
C_i , O_i, M_i	The CPU, I/O and Memory of U_i
L_j	Size of task T_j
D_j	Deadline of task T_j
C_j, O_j, M_j	CPU, I/O and Memory of task T_j
C_s, O_s, M_s	CPU, I/O and Memory of the system
T_{jh}	The category of task T_j
C, O, M	The rate of C_j, O_j, M_j and C_s, O_s, M_s
Q_C, Q_O, Q_M	Resources queues of CPU, I/O and Memory
Q_{TC}, Q_{TO}, Q_{TM}	Task queues of CPU, I/O and memory

3.1 Basic Terminologies

In cloud computing, a scheduling model is assumed there are N resources U = {u1, u2,…,ui, . . . , uN} and K tasks {T1, T2, . . . ,Tj, . . . , TK}.

3.1.1 *Resources:* A resource is a virtual machine defined by the values CPU, I/O, and memory i.e. Ui = (Ci, Oi, Mi) this are CPU utilization, I/O waiting time, and memory usage respectively. Values of this variables are brought from the global resource manager, which periodically composes and brings information from local resource managers.

3.1.2 *Tasks:* Tj = (Cj, Lj, Mj, Dj), here Cj is CPU usage, Lj is task size, Mj is memory and Dj is deadline of task in which the task have to complete its execution. Task size is equal to the length of the task Information of these values taken from the task manager.

3.1.3 I/O usage: I/O usage of the task Tj is defined as Oj = Lj/Cj. Lj is the task size, and Cj shows the capacity of CPU to complete the task Tj.

3.1.4 *Resource Capacity*: The variables (Ci, Oi, Mi) denotes the resource capacity known for CPU, I/O, and memory. At the same time, It denotes tasks resource requirement of the CPU, I/O, and memory.

3.1.5 *Assumption 1:* The first assumption is information submitted by user is trustworthy, where the value for Lj can be provided correctly. Dj is the deadline in which the task will be complete execution. The variables (Cj, Mj) are the values obtained by the user and are enough to complete the task.

3.1.6 *Assumption 2:* Second assumption is that variable values are fixed and do not change during the lifetime of the task means it shoud have fixed intensity in its life time.

3.1.7 Assumption 3: Third assumption is (Ci, Oi, Mi) are true valued because resources are ambiguous in cloud computing.

3.2 System Architecture

This section describes system architecture of MIPSM includes three steps 1) task classification, 2)resource sorting & 3)interlacing peak scheduling.

3.2.1 Task Classification

In first step of task classification, task manager responsible for undertaking task request for arrival queue by users. The information about task CPU usage, task size and memory gives demand for resources. Task classification requires information regarding I/O usage is calculated using the task size and CPU capacity. It is necessary to know information of of C_s, O_s and M_s of the system about CPU, I/O and memory respectively before task classification. The task classification performs on the basis of formula 1, there it give one value among C,O and M for CPU, I/O and memory respectively. The largest value denotes task category T_{jh} among CPU, I/O, memory queues.

$$\text{Tjh} = \max(C, O, M) = \max\left(\frac{Cj}{Cs}, \frac{Oj}{Os}, \frac{Mj}{Ms}\right) \quad (1)$$

If Suppose the T_{jh} has value 0 then that task is considered to be I/O intensified, the queues for K tasks are Q_{TC}, Q_{TO} and Q_{TM} of a for CPU intensified, b for I/O intensified, K-a-b memory intensified by the task category T_{jh} as follows:

$$Q_{TC}:\{T_1, T_2,\ldots, T_{jc},\ldots,T_a\} \quad (2)$$

$$Q_{TO}: \{T_{a+1}, T_{a+2},\ldots,T_{jo},\ldots,T_{a+b}\} \quad (3)$$

$$Q_{TM}: \{T_{a+b+1}, T_{a+b+2},\ldots,T_{jm},\ldots,T_{k-a-b}\} \quad (4)$$

3.2.2 Resource sorting

In this second step, resource manager responsible for gathering information from global resource manager, which is bring from local resource manager, then global resource manager sorts all resources in ascending order for three Q_C, Q_O, and Q_M as follows:

$$Q_C = \{U_1, U_2, \ldots\ldots\ldots\ldots, U_{ic}, \ldots\ldots\ldots\ldots, U_N\} \quad (5)$$

$$Q_O = \{U_1, U_2, \ldots\ldots\ldots\ldots, U_{io}, \ldots\ldots\ldots\ldots, U_N\} \quad (6)$$

$$Q_M = \{U_1, U_2, \ldots\ldots\ldots\ldots, U_{im}, \ldots\ldots\ldots\ldots, U_N\} \quad (7)$$

3.2.3 Interlacing Peak Scheduling Method

The Interlacing peak scheduling method is MIPSM as shown in figure 2. In first step task manager undertaking all task for classification of K task, which was done on the basis of task request of resources for execution. Then that all K tasks are classified into three queues: 1) CPU intensive, 2) I/O intensive and 3) memory intensive. Then in next step, global resource manager which composes and brings information from local resource manager. Local resource manager also composes and brings information about task demand for resources from local nodes. Then in the Last global resource manager performs sorting of all resources in ascending order as shown in figure 3. As per demands there are two kinds of queues i.e task queue and resource queue. Then in final step the scheduling method used to allocate task to demanded resources. It implicate the peak of resource uasage.

It implicates the peak resource usage according to resource demands of tasks. In the first Nature of task is determined according to formula 1, Then if the value of task category obtained C then task is considered to be CPU intensified i.e. that task demands for low CPU capacity then task goes for CPU intensive queue, then if the value of task category obtained O then task is considered to be I/O intensified i.e. that task demands for low I/O operation then task goes for I/O intensive queue, Then if the value of task category obtained M then task is considered to be memory intensified i.e. that task demands for low memory requirement then task goes for memory intensive queue.

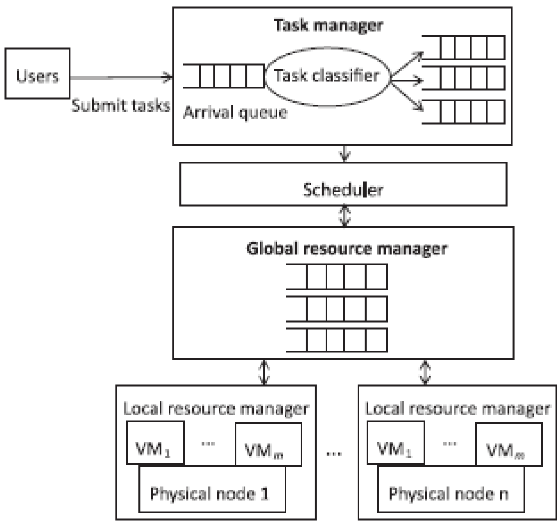

Figure 2: Interlacing Peak Scheduling Method

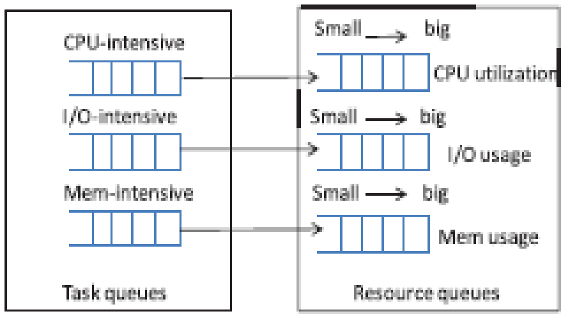

Figure 3: Task and Resource queues

Algorithm: Multiqueue Interlacing Peak Scheduling Method

Input: $T_1, T_2, \ldots\ldots T_j, \ldots\ldots T_k, T_j = (C_j, L_j, M_j, D_j), Q_c, Q_o, Q_m, C_j, O_J, M_j, C_s, O_s, M_s$

Output: (T_j, Q_x)

1. BEGIN
2. FOR j=1 to k
3. Calculate O_j using Definition 3;
4. Calculate T_{jh} using Formula 1;
5. IF T_{jh} = C Then
6. x = C;
7. $T_j \rightarrow Q_c$;
8. END IF
9. IF T_{jh} = O Then
10. x = O;
11. $T_j \rightarrow Q_o$;
12. END IF
13. IF T_{jh} = M Then
14. x = M;
15. $T_j \rightarrow Q_m$;
16. END IF
17. END FOR
18. END

The interlacing peak scheduling method balances load according two strategies that are as follows:
Case 1: when $N \geq K$, task are scheduled according to algorithm as follows:

$$Q_{TC} \rightarrow \{U_1, U_2, \ldots, U_a\} \quad (8)$$

$$Q_{TC} \rightarrow \{U_1, U_2, \ldots, U_{a+b}\} \quad (9)$$

$$Q_{TM} \rightarrow \{U_1, U_2, \ldots, U_{k-a-b}\} \quad (10)$$

Case 2: when $N<K$ No. of resources are less than the No. of task, so the task are scheduled in groups, First ($h= [\frac{N}{3}]$) tasks are chosen for queue allocated according to algorithm as follows:

$$Q_{TC} \rightarrow \{U_1, U_2, \ldots, U_h\} \quad (11)$$

$$Q_{TO} \rightarrow \{U_1, U_2, \ldots, U_h\} \quad (12)$$

$$Q_{TM} \rightarrow \{U_1, U_2, \ldots, U_h\} \quad (13)$$

Then in next group K-h tasks are scheduled, If $K-h < N$, then the $[\frac{K-h}{3}]$ task will be allocated to the three queues ; otherwise $[\frac{N}{3}]$ task are scheduled, then remaining K-2h tasks are allocated in next groups.

4. SIMULATION OF CLOUDSIM

Cloudsim is open source toolkit that performs simulation of cloud computing using its generalized framework developed in java. Cloudsim toolkit developes that support for modeling and simulation of large scale Cloud computing environments, including data centers, on a single physical computing node.

4.1 Parameter Description

In this cloud computing system, there are 100 host and 10 virtual machines on each host of having CPU computing ability in the range 1860 MIPs to 2660 MIPs having disk I/O 10 GB and RAM 4096 MB, the setup is as shown in Table 2

Table 2: VM Parameter setup in each host

Parameter	Value
CPU computing ability	1860 MIPs,2660 MIPs
Disk I/O	10 GB
RAM	4096 MB
Bandwidth	100 M/s
Storage	10G

The task setup of data center is as shown in Table 3

Table 3: Task setup of Data Center

Parameter	Value
Length (CPU)	[400,1000] MIPs
File Size	[200,1000] MB
Output size (memory)	[20,40] MB

4.2 Simulation Description

Result analysis was conducted on Dell PC with Intel i3 CPU and 2 GB of memory running window 7 and Cloudsim 3.0. Cloudsim is used to 10 virtual machines in single data center. This system having task classification method contains the number of tasks and these tasks are provided for classification. When we create the task that time applying MIPSM algorithm achieve higher average response time.

The response time of task is calculated as the time at which task is submitted for scheduling till completion of task execution. The average response time is calculated from formula 14

$$t_{response} = \frac{\sum(t_{j-wait}+t_{j-complete})}{K} \quad (14)$$

Figure 4 states comparison of average response time of MIPSM and average response time of FCFS for 500 task. The task arrival rates were 40,60,80,90 tasks/s. In MIPSM average response time increases as the task arrival rate increases.

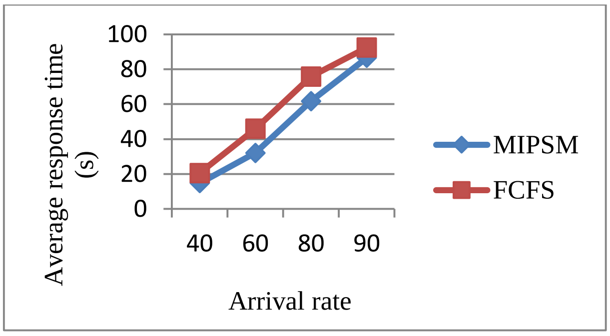

Figure 4: Average response time for MIPSM and FCFS for 500 tasks

Then figure 5, states comparison of average response time of MIPSM and average response time of FCFS for 1200 task. Next experiment verifies task classification through load balancing, in this 10 random resources were selected randomly and then observed their CPU, I/O and Memory usage. Figure 6, states comparison of CPU utilization of MIPSM and CPU utilization of FCFS for 500 tasks. Figure 7, states comparison of CPU utilization of MIPSM and CPU utilization of FCFS for 1200 tasks. Fluctuations of CPU usage are too obvious for FCFS , however fluctuations in MIPSM were very less that had balanced load effectively

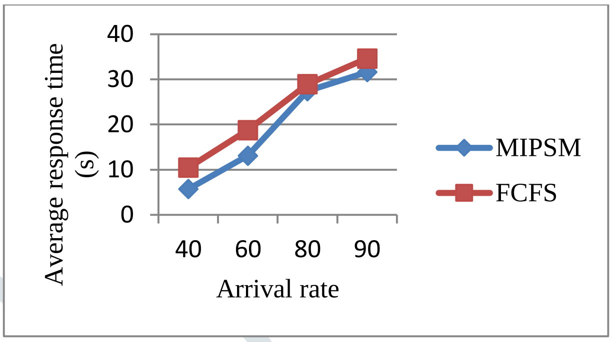

Figure 5:Average response time of MIPSM and FCFS for 1200 task

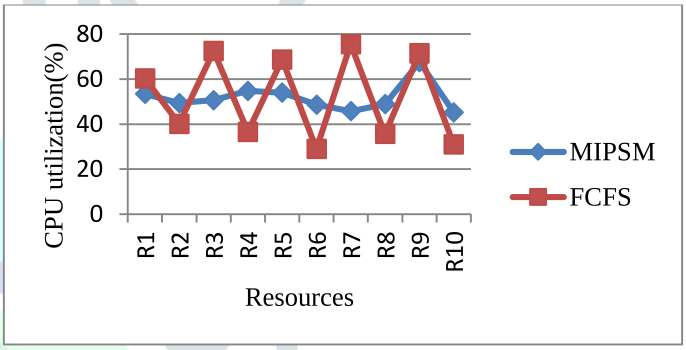

Figure 6:CPU Utilization of MIPSM and FCFS for 500 task

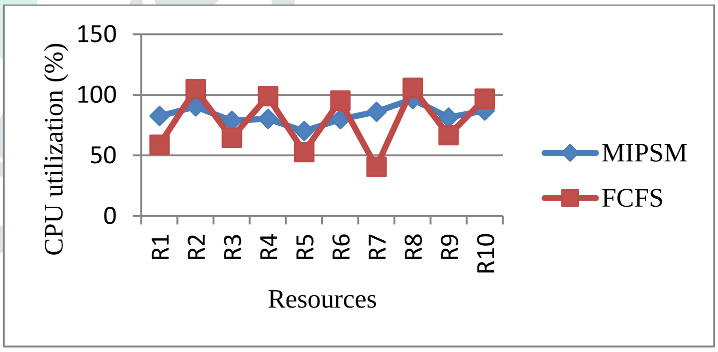

Figure 7: CPU Utilization of MIPSM and FCFS for 1200 task

In Next Figure 8, it shows comparison of I/O utilization for MIPSM and FCFS for 500 task.

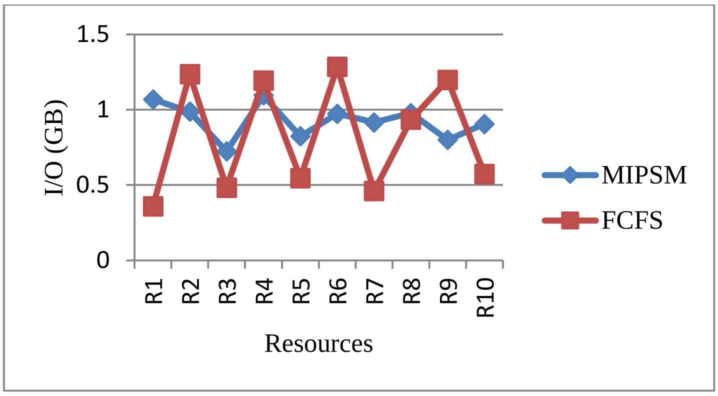

Figure 8: I/O Utilization of MIPSM and FCFS for 500 tasks

Figure 9, it shows comparison of I/O utilization for MIPSM and FCFS for 1200 task, here also The fluctuation in resources were too fewer for MIPSM as compared to FCFS.

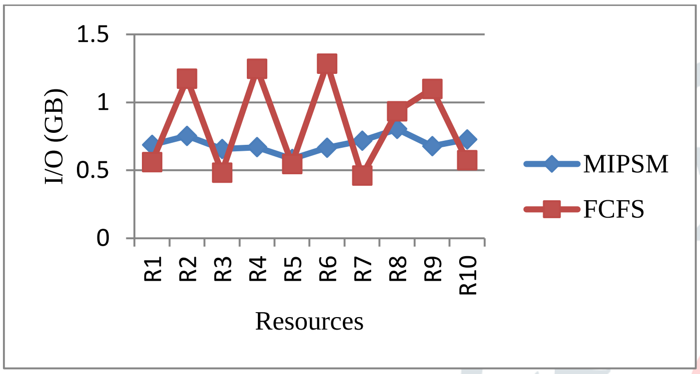

Figure 9: I/O Utilization of MIPSM and FCFS for 1200 tasks

In Next Figure 10, it shows comparison of memory utilization for MIPSM and FCFS for 500 task.

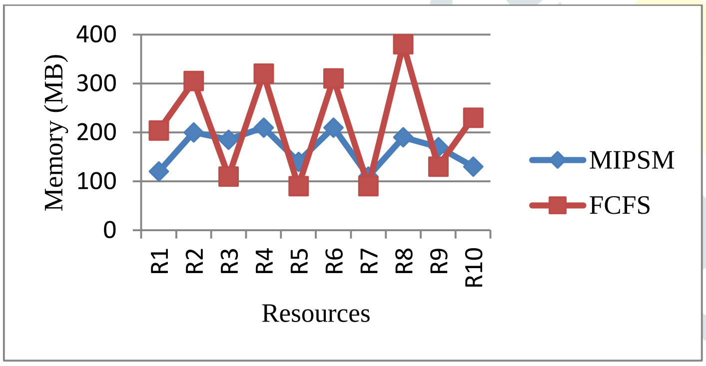

Figure 10:Memory utilization of MIPSM and FCFS for 500 tasks

Figure 11, it shows comparison of memory utilization for MIPSM and FCFS for 1200 task, here also The fluctuation in resources were too slight for MIPSM as compared to FCFS.

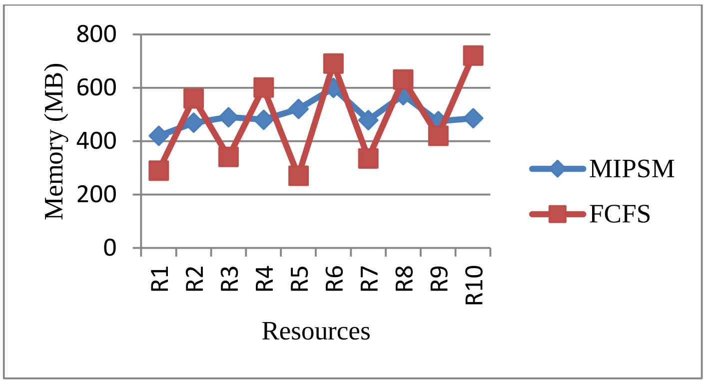

Figure 11: Memory utilization of MIPSM and FCFS for 1200 tasks

In Next Figure 12, it shows comparison of resource utilization for MIPSM and FCFS for 500 task. Figure 13, it shows comparison of resource utilization for MIPSM and FCFS for 1200 task, here also The resource utilization for ten resources showed little difference for MIPSM as compared to FCFS.

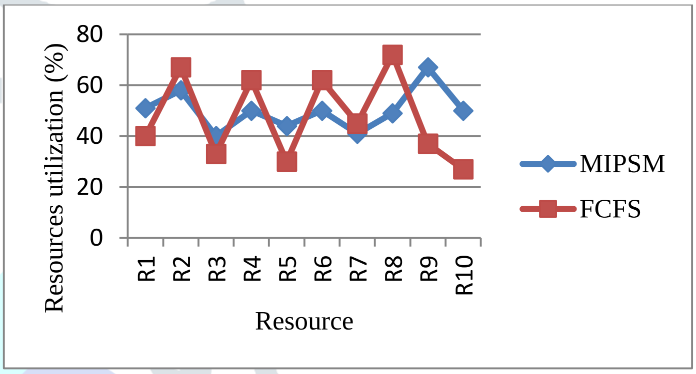

Figure 12: Resource utilization of MIPSM and FCFS for 500 tasks

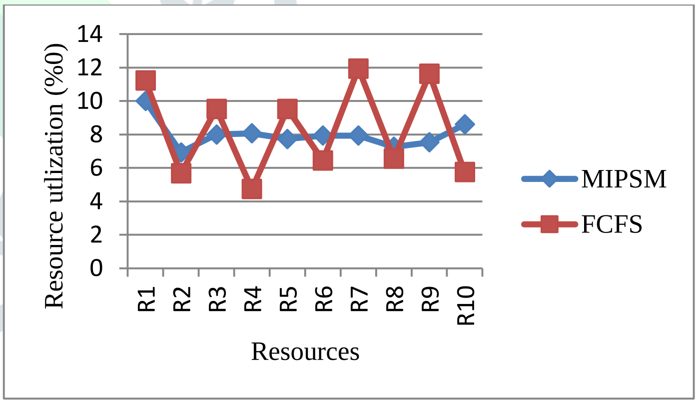

Figure 13: Resource utilization of MIPSM and FCFS of 1200 tasks

Deadline violation rate, if running time T_j is greater than deadline D_j then task is considered to violate the deadline constraint. The Deadline violation rate is calculated as formula (15).

$$\upsilon = \frac{n_d}{K} * 100\% \qquad (15)$$

Where n_d is the number of times deadline violated in K tasks. In Next Figure 14, it shows comparison of deadline violation rate for MIPSM and FCFS for 500 task. Figure 15, it shows comparison of deadline violation rate for MIPSM and FCFS for 1200 task, here deadline violation rate increases as task arrival rate increases for MIPSM but for FCFS deadline violation rate is very high as compared to MIPSM.

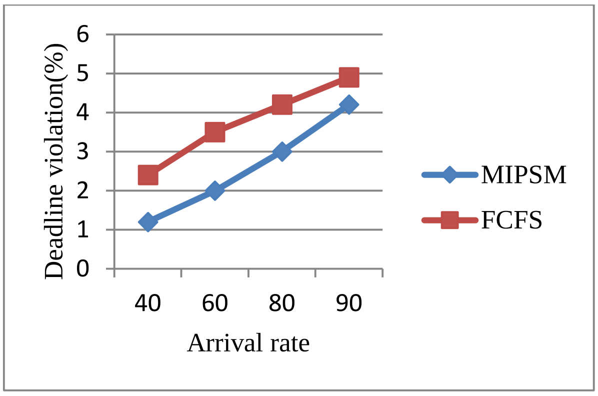

Figure 14: Deadline violation rate of MIPSM and FCFS for 500 tasks

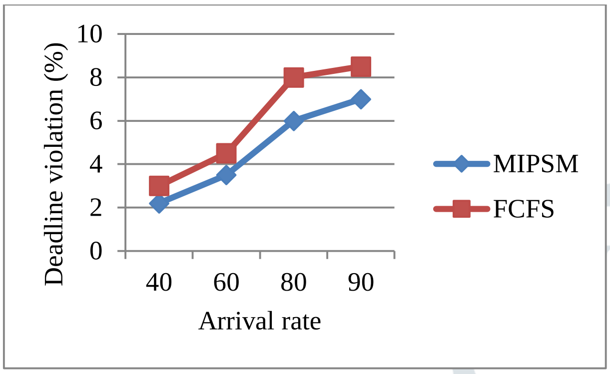

Figure 15: Deadline violation rate of MIPSM and FCFS for 1200 tasks

5. CONCLUSION

In cloud computing, diversity of task, dynamic factors of resource can dynamically changes over time, this could causes load imbalances and affect the performance and resource utilization. MIPSM method solve this issues in which firstly task are classified into three queues named CPU intense, I/O intensive and memory intensive, next step resources were sorted according to CPU utilization, I/O wait times, memory usage and in last, three queues of task were scheduled to those resources whose loads are lighter than others.

6. REFERENCES

[1] Shoubin Dong, L. Shu, Chunsheng Zhu, G. Han L. Zuo, "A Multiqueue Interlacing Peak Scheduling method based on Tasks Classification in Cloud Computing," *IEEE System Journal*, April 2016.

[2] R. Buyya, C. S. Yeoa, S. Venugopala, J. Broberga, and I. Brandicc, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer Systems*, vol. 25, no. 6, pp. 599-616, 2009.

[3] J. L. Hellerstein and R. Boutaba A. K. Mishra, "Towards characterizing cloud backend workloads; insights from google compute clusters," *ACM SIGMETRICS Perform. EVAL. Rev.*, vol. 37, pp. 34-41, March 2010.

[4] J. L. Hellerstein, R. Boutaba Q. Zhang, "Characterizing task usage shapes in Google's compute clusters," *Proc. Large Scale Distrib. Syst. Middleware Workshop (LADIS)*, pp. 1-6, 2011.

[5] X. Yan, S. Tao and N. Anerousis A. Khan, "Workload Characterization and prediction in the cloud: A multiple time series approach," *Proc. IEEE Netw. oper. Manage. Symp. (NOMS)*, pp. 1287-1294, 2012.

[6] M. F. Zhani, R. Boutaba, J. H. L. Hellerstein Q. Zhang, "HARMONY:Dynamic heterogenity? Aware resource provisioning in the cloud," *Proc. 33rd IEEE Int. Conf. Distrib. Syst.*, pp. 511-519, 2013.

[7] P. Garraghan, P. Townend and J. Xu. I. S. Moreno, "An approach for characterizing workloads in Google cloud to derive realistic resource utilization models," *Proc. 7th IEEE Int. Symp. Serv-Oriented Syst. Eng.*, pp. 49-60, 2013.

[8] A. Konwinski, M. Abd-El-Malek, J. Wilkes M. Schwarzkopf, "Omega:flexible,scalable schedulers for large compute clusters," *Proc. 8th ACM Eur. Conf. Comput. Syst.*, pp. 351-364, 2013.