# Deadline Aware Prioritized Task Scheduling Algorithm in Cloud Computing

**R. Barani, S. Suguna**

***Abstract--- Cloud Computing is a computing Paradigm in which services are provided by service providers on pay-per-use. Task Scheduling is the challenging issue in cloud computing. Task scheduling refers to allocating tasks to available resources to achieve better performance of the system. Here we have proposed a Heuristics algorithm to schedule tasks in given resources which satisfies the QoS of system taking Priority and Deadline of tasks as parameters. Our algorithm is compared with existing algorithms like EDF and TLD algorithms. Our algorithm provides better makespan, increases throughput and utilize resources well compared to existing algorithms.***

***Keywords--- Task Scheduling - TLD Algorithm –EDF Algorithm –Proposed Algorithm –Comparative Analysis***

## I. INTRODUCTION

Cloud computing is the computing service delivered to the user by providers. Cloud providers provide different types of service like Software as a Service(SaaS) where application software and database access are provided. Platform as a Service(PaaS) where computing platform are provided and Infrastructure as a Service(IaaS) where Infrastructure is provided to user which includes processing power and storage of virtual machines.

These services are providedas a pay-per-use service. Task Scheduling refers to allocating tasks to available Virtual machines. A good task scheduling improves the performance of the system and help to achieve good Quality of Service(QoS).The objective of task scheduling is to minimize the makespan and maximize the Resource Utilization.Many Task scheduling algorithms were proposed but we need more improvement to make cloud system a better one.

In this paper we propose a Heuristics algorithm called Deadline Aware Prioritized Task Scheduling Algorithm(DPTS)to schedule tasks in available Virtual machines. In Section 2,we provide the literature survey of available algorithms for task scheduling. In section 3,our proposed algorithm was explained. In Section 4, the comparative analysis of proposed algorithm with the existing algorithms are made and the results are tabulated. Finally we conclude about the merits and demerits of proposed algorithm.

## II. LITERATURE SURVEY

Omer Khalid et al[1] designed a deadline-aware algorithm which schedules a virtual machine for minimizing overhead and it is used to minimize job-deadlines when it is given to resources.Patel et al[2] proposed an improved priority based scheduling using iterative methods. This algorithm has better makespan and consistency than other algorithms. Neelimapriyanka et al[3] proposed an Optimal priority based EDF scheduling algorithm for allocating resources using task priorities, deadlines and its processing times.It shows better performance of turnaround time, waiting time and deadline violation when compared to FCFS,SJF and priority based EDF scheduling.

DeeptikaSaxena et al[4] proposed a dynamic fair priority optimization task scheduling algorithm in which they classified and grouped all tasks based on their deadline and cost constraints. Here different priority queues (high, mid, low) are implemented in round robin fashion. This algorithm provides better efficiency and fairness at system level. NeelimaVarma et al[5] presented Round robin based Prioritized Earliest Deadline first scheduling model based on time quantum,priorities and deadlines. It is compared with traditional algorithms like FCFS and SJF which reduces the turnaround time,waiting time and average deadline violation.

Pankajkumar Chauhan et al[6] proposed a model in which they have assigned a rank for each task and scheduled using min-min strategy. This algorithm is advantageous in terms of task scheduling based on task length and deadline. K.Raja et al [7] proposed an improved credit basedalgorithm which uses the parameters like userpriority, task length and deadline. This algorithm reduces makespan of tasks. Antony Thomas et al [8] proposed a Credit based scheduling algorithm which is based on cloudlet length and priority. In this makespan is decreased and shows improvement in resource Utilization.

## III.EXISTING METHODOLOGIES

Garima Gupta et al [9] present a priority based earliest deadline first algorithm with task migration. They allocate priority to task based on their deadline and allocate resources to them. The tasks are migrated when they missed their deadline. This improves overall response time and gives better performance.In this algorithm the tasks are allocated only on deadline value.Waiting queue is introduced for pre-empted tasks and processed later based on their priority. This increases the overhead of the system.

Arnav Wadhonkaret al[10] proposed an algorithm which allocates tasks based on their length and deadline.This algorithm has two scheduling criteria. First is task length and other is deadline of task. It is compared with algorithm which is based on task length and algorithm which is based on deadline.

**Revised Version Manuscript Received on 16 September, 2019.**

**R. Barani,** Assistant Professor, Department of Computer Science, Annai Women’s College, Karur. (e-mail: baraniraj77@gmail.com)

**S. Suguna**, Assistant Professor, Department of Computer Science, Sri Meenakshi Govt. Arts College for Women (A), Madurai.
(e-mail: kt.suguna@gmail.com)

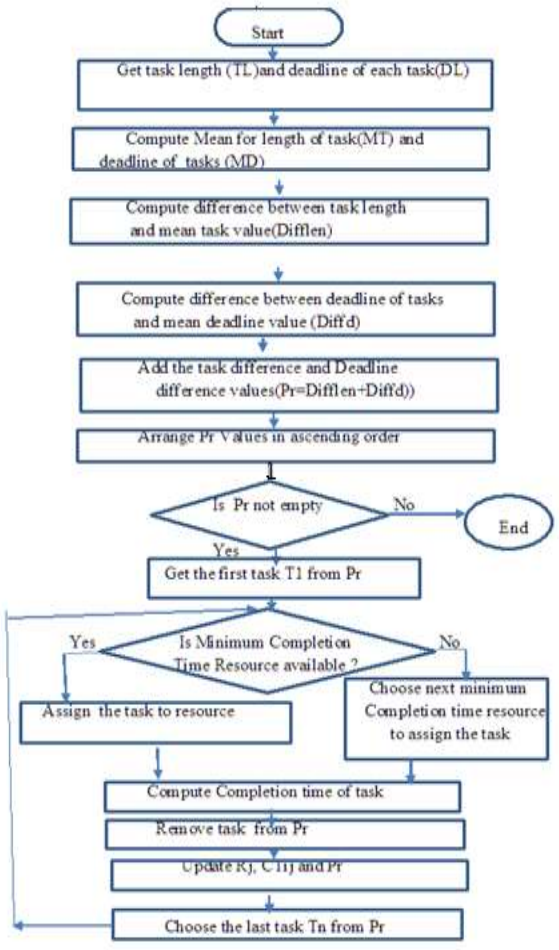
The presented algorithm shows reduction in makespan and average waiting time than compared algorithms. We have proposed atask scheduling algorithm which takes two parametersDeadlineand Task length to improve the performance of the system.

### *3.1 Proposed Methodology*

In our algorithm we have considered two constraints. One is task length and other is Deadline. First we are taking the mean for task length and finding the difference between mean value and task length of each task. In the same way, we are finding the mean for deadline value and finding the difference between the mean value and deadline value of each task. Then we are adding the two difference values of task and deadline. These values are arranged in ascending order. We select the first value(low value) from list and assign it to minimum completion time resource if it is available. Then the last value(high value) is selected and assigned to next minimum completion time resource available. This process is repeated until the list is empty.

As the list is arranged based on difference value of mean and we are choosing the low value and high value in the list alternatively, starvation of tasks are reduced.

The flowchart for this algorithm is shown inFigure 1



Start
Get task length (TL)and deadline of each task(DL)
Compute Mean for length of task(MT) and deadline of tasks (MD)
Compute difference between task length and mean task value(Difflen)
Compute difference between deadline of tasks and mean deadline value (Diffd)
Add the task difference and Deadline difference values(Pr=Difflen+Diffd))
Arrange Pr Values in ascending order
Is Pr not empty
No
End
Yes
Get the first task T1 from Pr
Is Minimum Completion Time Resource available ?
Yes
No
Assign the task to resource
Choose next minimum Completion time resource to assign the task
Compute Completion time of task
Remove task from Pr
Update Rj, Cij and Pr
Choose the last task Tn from Pr

**Figure1: Flowchart of Proposed Algorithm**

We calculate the Average throughput of the system as
*Average Throughput=No of tasks/makespan*
and Average Utilization of resources as
Average Resource Utilization=

*∑time taken by resource to finish all tasks/(makespan*n),n=Number of resources*

The pseudocode of the proposed algorithm (DPTSA) is shown in Algorithm1.

**Algorithm 1:Pseucode of Proposed Algorithm**

```
Input: 'n' number of job request Ti with its priority Pi and deadlineDLi.
 Number of VMs (ie Resources) Rj to process request
Output: Mapping of Tasks in resources considering both Task length as priority and deadline of job.
------------------------------------------------------------------------------------------------------------
1.  Get each task length (TLi)
2.  Get Deadline of each tasks (TDi)
3.  Calculate the mean for length of task as MT=∑TLi, 1<=i<=n and
4.  Calculate mean for deadline of tasks MD= ∑ TDi, 1<=i<=n
5.  Compute Diffleni= │MT-TLi│ and
6.  Compute Diffdi= │MD-TLi│
7. Add both task length difference value and deadline difference values.
Pri=DiffLeni+Diffdi
8. Sort Pri in ascending order
9.Repeat the following steps until no tasks isavailable in pri.
10.Check if Minimum completion time resource is available.
11.If yes then
    11.1 Get the I task from the List(pri=lowvalue).
    11.2  Assign task Ti to  Resource $R_j$
    11.3. Compute completion time of task asCij=ETij+Wti
    11.4 Remove the task $T_i$ from task list.
    11.5   Update $R_j$&Cij.
12. Next choose the task Tnfrom list(pri=high value)
13. Repeat the steps 11.2 thro 11.5.
14. Alternatively choose the minimum valuefrompri and then maximum value from pri for next iteration.
15. if Minimum completion time resource notavailable then
16. Choose next available minimum completion time resource to allocate the tasks.
17. If resourcenot available, wait in queue until resource becomes free.
```

## IV.RESULTS AND DISCUSSION

The algorithm is simulated in simulation environment cloudsim 3.0.3.The simulation is done under following conditions.

**Table1: Basic Configuration**

| | |
|---|---|
| Number of Datacenters | 2 |
| Number of Cloudlets | 10 |
| Number of brokers | 1 |
| Number of Hosts under Datacenter | 2 |

Each datacenter has several hosts. Host configuration is shown as

**Table 2: Host configuration**

| | |
|---|---|
| RAM(MB) | 16384 |
| MIPS(Lines of Codes) | 1000 |
| Storage(MB) | 1000000 |
| Bandwidth(MB/sec) | 10000 |
| Number of Virtual Machines | 3 |

Hostconsists of several virtual machines. Each virtual machine has own configuration. Here we consider the same configuration is applied for each VMs varying its processing speed. The configuration of virtual machine are shown as

**Table 3:Virtual Machine Configuration**

| | |
|---|---|
| Number cores | 2 |
| MIPS (Lines of Codes) | 1000 |
| Size(MB) | 10000 |
| RAM(MB) | 512 |
| Bandwidth(MB/sec) | 1000 |

We are taking 7 tasks whose task length are (2400,8200,5600,4200,1600,3200,2500) and their deadline (102,151,194,118,57,69,72). We are allocating these 7 tasks in three resources whose processing speed are (300,100,50). The Makespan of TLD algorithm is 120, EDF algorithm is 114 whereas proposed algorithm is 82.This shows the makespan is reduced in proposed algorithm.TheComparison table of algorithms for executing 20 tasks in 3 resources is as below

**Table 4: Comparison table of algorithms**

| Metrics /Algorithms | TLD | EDF | Proposed(DPTSA) |
|---|---|---|---|
| Makespan | 42.6 | 34 | 32 |
| Throughput | 0.47 | 0.59 | 0.63 |
| Resource Utilization | 77% | 91% | 94% |

From the above table it is clear that the makespan is reduced and throughput is increased. From the provider point the resource utilization is high compared to other algorithms.

The comparison chart for makespan when executing tasks in 3 resources are shown in figure 2.

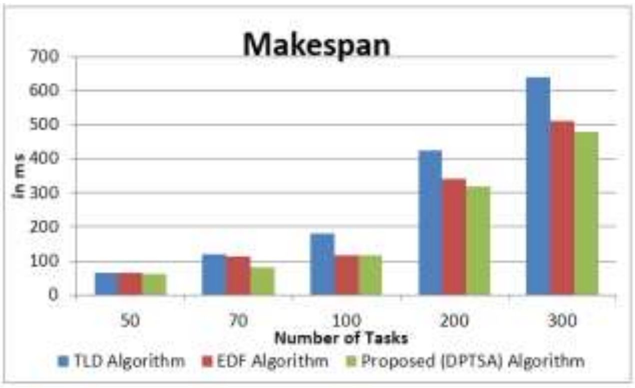


**Figure 2: Comparison chart of makespan**

This shows that the makespan is reduced in proposed algorithm.

The comparison chart for resource utilization is shown in figure 3.

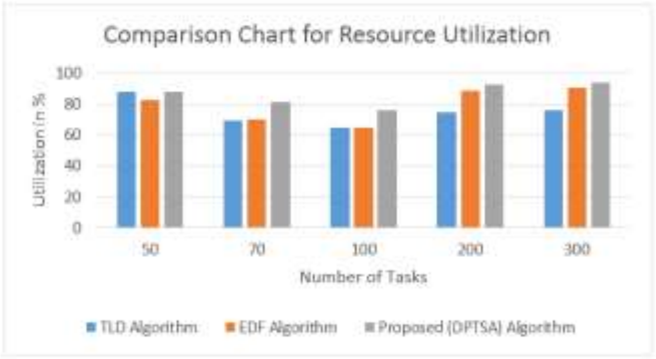


**Figure 3: Comparison chart of Resource Utilization**

The above chart shows that the resource utilization is high in proposed algorithm.

The comparison chart of throughput is shown in Figure 4 as follows

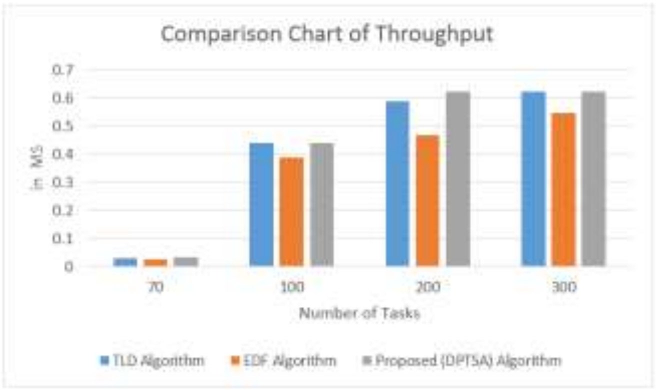


**Figure 4: Comparison chart of Throughput**

The chart shows that the throughput is increased in proposed algorithm when compared to existing algorithm.

## V. CONCLUSION

The Efficient Scheduling of Tasks to available resources will improve performance of the system. Users desire to reduce makespan of executing tasks and providerdesire to increase Resource utilization. In our proposed algorithm we have increased the resource utilization on the average of 15.46% than compared algorithms and makespan is reduced on an average of 23.04%.Throughput also increased on an average of 19.04% when compared to other existing algorithms like TLD and Prioritized EDT algorithms. Thus the proposed algorithm improves the performance by an average of 19.18% than existing algorithms.In future we consider the workflow tasks and work to reduce the time latency.

## REFERENCES

[1] Omer Khalid, Ivo Maljevic, Richard Anthony, Miltos Petridis, Kevin Parrott and Markus Schulz (2010), “Deadline Aware Virtual Machine Scheduler for Grid and Cloud Computing”, *Proceeding of 24th IEEE International Conference on Advanced Information Networking and Applications Workshops*, pp 85-90.

[2] Patel, Swati J, and Upendra R. Bhoi. "Improved Priority Based Job Scheduling Algorithm in Cloud Computing Using Iterative Method." Advances in Computing and Communications (ICACC), 2014 *Fourth International Conference on. IEEE*, 2014.

[3] Neelima Priyanka, Suresh Varmaa, Krishnam Raju Indukuri and Rama Sundari, "Optimal Priority based Earliest Deadline Scheduling in cloud Computing",International Journal of innovations and Advancement in Computer Science,ISSN 2347-8616,Volume 6,Issue 11,November 2017.

[4] Deepika Saxena, Dr.R.K. Chouhan, Dr.RamesKait," Dynamic Fair Priority Optimization Task Scheduling Algorithm in Cloud Computing: Concepts and Implementations", *I.J. Computer Network and Information Security*,2016,2,41-48.

[5] Neelima Priyanka N, Suresh Varma P, R Krishnam Raju Indukuri, B Sukumar Babu, "Round Robin based Prioritized Earliest Deadline First Scheduling in Cloud Computing", *International Journal of Engineering Research in Computer Science and Engineering,* Vol 4,Issue 11, November 2017.

[6] Pankajkumar Chauhan, Payal Jaglan and Poonam Dabas, "An Intensify Deadline Aware Credit Based Cloud Task Scheduling", *International conference on Computing, Communication and Automation (ICCCA 2016), IEEE* 2016.

[7] K.Raja and G.Sekar, "An Algorithm for Credit Based Scheduling in Cloud Computing Environment depending upon Deadline strategy", *International Journal of Current Trends in Engineering and Research(IJCTER)*,volume 02,Issue 08,August-2016.

[8] Antony Thomas, Krishnalal G and Jagathy Raj V P,"Credit Based Scheduling Algorithm in Cloud Computing Environment",*International Conference on Information and Communication Technologies(ICICT 2014), procedia computer science* 46(2015) 913-920.Elsevier.

[9] Garima Gupta, Vimal Kr.Kumawat, P R Laxmi, Dharmendra Singh Vinesh Jain, Ravinder Singh, "A simulation of Priority based Earliest Deadline First Scheduling for Cloud Computing System",2014 *First International Conference on Networks & Soft Computing, IEEE.*

[10] Arnav Wadhonkar and Deepti Theng, "A Task Scheduling Algorithm based on Task Length and Deadline in cloud computing", *International Journal of Scientific and Engineering Research*, Volume 7, Issue 4, April 2016.