HYMM: A New Heuristic in Cloud Computing

Neha Sharma[1], Dr. Sanjay Tyagi[2], Swati Atri[3]

[1]*M.Tech. Scholar, Department of Computer Science & Applications, Kurukshetra University, Haryana, India*
[2]*Assistant Professor, Department of Computer Science & Applications, Kurukshetra University, Haryana, India*
[3]*Ph.D. Scholar, Department of Computer Science & Applications, Kurukshetra University, Haryana, India*

---***---

Abstract - *Cloud computing is getting popular day by day. With increase in popularity, the computational requirement of computing has also increased. For effective and efficient cloud computing environment, one major requirement is scheduling of tasks. For effective computing, various task scheduling algorithms have been developed so far. This paper introduces a new scheduling algorithm based on two traditional yet conventional algorithms known as Min-Min and Max-Min. This proposed algorithm utilizes the advantages of both and tries to overcome the disadvantages of them. For evaluating this work, a simulator known as CloudSim has been used. Then the proposed algorithm has been compared based on makespan, average resource utilization and load balancing factor with the existing conventional heuristic algorithms. Results and Analysis show that the proposed algorithm outperforms the Min-Min and Max-Min heuristic algorithms.*

Keywords- Cloud Computing, Heuristic Algorithm, Max-Min Algorithm, Min-Min Algorithm, Task Scheduling.

I. INTRODUCTION

In present era, cloud computing is developing day by day as it is focusing on reducing cost for communication, reducing average time needed in transactions etc. Thus cloud computing has a potential to effect the daily life routine of people. Cloud computing is different from traditional computing because of its large scale computing power, cost effectiveness, Service Level Agreement(SLA) between user and provider etc. [1]. Cloud computing environment is vast and complex for handling number of jobs arriving per millisecond. Cloud computing is now becoming a commercial platform for many business giants, so assigning number of resources to the requests, coming from the users is a critical task. Here the need for task scheduling arises. Task Scheduling in cloud computing environment is a NP-hard problem. Many task scheduling algorithms have been proposed to efficiently and effectively schedule the tasks [2] [3].

Assigning a job to the best available resource is known as task scheduling. In a cloud, there are heterogeneous structuring elements and scheduling in such heterogeneity is a major challenge [4].

In cloud computing there is mainly two type of scheduling:

a) Online mode scheduling:

In this type of scheduling, the job is assigned to the resource as soon as it arrives. Minimum Execution Time (MET) and Minimum Completion Time (MCT) are examples of online scheduling [5].

b) Batch mode Heuristic Offline scheduling:

In this type of scheduling, the jobs are collected as they arrive. The set where these jobs are collected is called the MetaTask (MT). Then the jobs in MT get scheduled on each resource as per the applied heuristic [6] e.g. Min-Min, Max-Min, RASA, Suffrage etc.

Scheduling in a cloud environment has three main phases: firstly, the available resource is discovered from the available pool of resources. Secondly, the best resource from the available resource is selected, and in last, the job or cloudlet is submitted to the selected resource [7].

In a cloud computing environment, an efficient and effective task scheduling algorithm is required [8]. A scheduling algorithm must satisfy some important performance metrics such as Makespan, better resource utilization, better Quality of Service (QoS) and better load balancing among resources etc [9].

In this paper, a scheduling algorithm named as Hybrid of Min-Min and Max-Min (HYMM) has been proposed. As its

name suggests, it is a scheduling algorithm, which utilizes the advantages of two prominent conventional scheduling algorithms (Min-Min and Max-Min) and at the same time overcomes their disadvantages. It gives better Makespan, Load Balancing and Resource Utilization in most of the cases when compared with Min-Min and Max-Min.
The remaining paper has been organized as follows: Section II gives the brief introduction of two basic task scheduling algorithms. Section III presents the introduction of proposed algorithm (HYMM). Next section i.e. section IV shows the simulation & analysis of proposed algorithm and its comparison with the existing scheduling algorithms, this section is followed by section V, which concludes the paper.

II. TASK SCHEDULING ALGORITHMS

Min-Min and Max-Min are two fundamental scheduling algorithms in grid computing, but they also performed better in cloud computing environment [10]. Both of these use the Minimum Execution Time (MET) and Minimum Completion Time (MCT) for allocating the tasks to the appropriate resource [7]. These are batch mode scheduling algorithms.

a) Min-Min Scheduling Algorithm:

Min-Min task scheduling algorithm has two phases, in first phase minimum completion time (MCT) of each task in metatask (MT) is calculated and in second phase, task which has the minimum completion time is executed first on the available fastest resource. This process is repeated until the Metatask (MT) gets empty [11] [12].
In Min-Min scheduling algorithm, the task having shorter length (MCT) is executed first and then the task having longer length (MCT) is executed. Thus the longer length tasks have to wait until all the shorter tasks get executed; this leads to greater makespan, which affect the cloud performance. The situation gets worse when the number of longer tasks is short [6] [1].

b) Max-Min Scheduling Algorithm:

In Max-Min task scheduling algorithm, the problem of Min-Min scheduling algorithm has been removed. It also has two phase long process but the change is only in second phase i.e. in allocating the tasks. In first phase, minimum completion time (MCT) of each task in metatask (MT) is calculated and in second phase, task which has the maximum MCT is executed first instead of the minimum MCT. This process is repeated until the Metatask (MT) gets empty [13] [14].
In Max-Min task scheduling algorithm, the tasks having shorter MCT get executed concurrently with the Long length task. It performs better when the number of short length tasks is high [15].

III. PROPOSED ALGORITHM

Min-Min and Max-Min scheduling heuristics have some advantages and disadvantages. Min-Min Scheduling gives poor performance when the numbers of long task length are very less as compared to short length tasks and Max-Min Scheduling cannot perform better when short length tasks are less. Thus in some cases, they produce larger makespan, do not provide resources effectively and at the same time leads to load imbalance [16] [17].
To address this problem, a new heuristic has been proposed named as Hybrid of Min-Min and Max-Min (HYMM) algorithm. This algorithm performs better in most of the cases. It provides:

- Better makespan i.e. total completion time is reduced.
- Efficient resource utilization as in Min-Min, best resource was always busy and other resources remain idle [18].
- Better Load balancing because inappropriate resource allocation generates imbalance load situations [9].

HYMM uses the advantages of Min-Min and Max-Min and overcomes their drawbacks. First, it calculates the Minimum Completion Time (MCT) of each task in metatask (MT), and then an average of task's length (ATL) is calculated. After calculating ATL, it is compared with each task length (TL_i). Two empty lists are created, first list is known as Minimum Weighted Task List (WTL_{min}) and second list is known as Maximum Weighted Task List (WTL_{max}). If TL_i is shorter than ATL, that task will be submitted to WTL_{min}, otherwise to WTL_{max}. This process repeats until all the tasks in metatask (MT) has been compared and it gets empty. After completing this process, WTL_{min} which has all the short length tasks, is scheduled according to the Max-Min scheduling process and WTL_{max}, which has all the long length tasks, is scheduled according to the Min-Min scheduling process. This complete process can be illustrated by using the flow chart represented in fig.1.

HYMM uses that scenario of the Min-Min and Max-Min, in which they perform their best and at the same time tries to overcome their drawbacks. As Min-Min gives better results when number of long length tasks are high, so WTL_{max} is scheduled according to this and Max-Min gives better results when number of short length tasks are high, so WTL_{min} is scheduled according to Max-Min schedular. Hence HYMM produces better results than both of the fundametal algorithms.

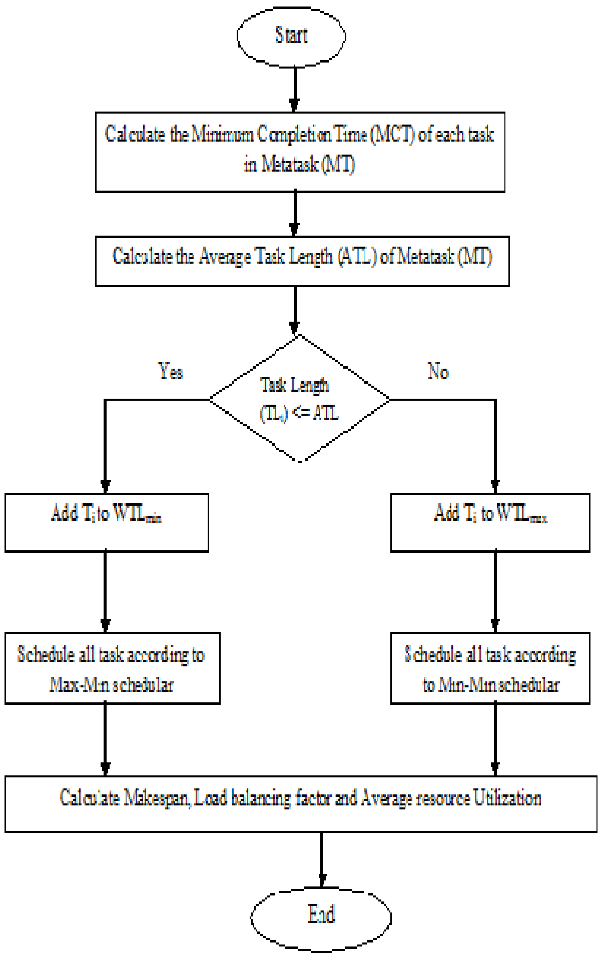
Start
Calculate the Minimum Completion Time (MCT) of each task in Metatask (MT)
Calculate the Average Task Length (ATL) of Metatask (MT)
Task Length (TLi) <= ATL
Yes
No
Add Ti to WTLmin
Add Ti to WTLmax
Schedule all task according to Max-Min schedular
Schedule all task according to Min-Min schedular
Calculate Makespan, Load balancing factor and Average resource Utilization
End

Figure 1: Flow Chart of Proposed Algorithm (HYMM)

- **Pseudo Code for Hybrid of Min-Min and Max-Min (HYMM) Algorithm:**

```
1.  For all submitted task ($t_i$) in Meta-Task (MT)
2.  For all resources ($R_j$)
3.  $CT_{ij}=ET_{ij} + r_j$
4.  End
5.  Calculate the Average Task Length (AVL)=
        $ATL = \sum_{i=1}^{n} TaskLength(TLi)/n$
6.  While MetaTask (MT) is not empty
7.  Do
8.  If (task length ($TL_i$) <= ATL)
9.  Add Task ($t_i$) to Minimum Weighted Task List ($WTL_{min}$)
10. Else
11. Add Task ($t_i$) to Maximum Weighted Task List ($WTL_{max}$)
12. End
13. Schedule the Minimum Weighted Task List ($WTL_{min}$) on Max-Min scheduler
14. Schedule the Maximum Weighted Task List ($WTL_{max}$) on Min-Min scheduler
15. End
16. End.
```

Here, t_i represents the task present in the Meta-Task. R represents the number of resources and r defines a particular resource. CT_{ij} has been used for completion time of task i on resource j, and ET_{ij} defines the execution time of task i on resource j.

IV. SIMULATION AND ANALYSIS

For simulating the proposed algorithm in a simulated or virtual environment, a simulation tool known as CloudSim has been used [19]. This tool has several advantages over other such type of tools. Some decent features of CloudSim are:

- It provides easy modeling of virtualized resource configuration, used in simulation, e.g. their RAM size, bandwidth, mips rate etc.
- It is capable for supporting large-scale Simulation experiment.
- It provides no upper limit for number of resources and tasks used in simulation process.

- There is no overhead for memory consumption [20].

Now a result analysis has been presented, where the performance is evaluated with respect to three performance metrics, they are:

a) Makespan: It is the total completion time in which a metatask get scheduled or executed. For effective performance of cloud, makespan should be low [21].

b) Average Resource Utilization: It can be defined as a measure of utilizing each resource present in cloud. For effective performance of cloud, average resource utilization factor should be high.

c) Load Balancing Factor: Load balancing in cloud scheduling is a major issue of concern. Due to improper resource utilization, load on each resource gets unbalanced [8]. For effective performance of cloud, Load Balancing factor should be high.

Based on these parameters, a performance analysis is shown below. Here, the proposed algorithm has been compared with Min-Min and Max-Min algorithms. In this scenario, number of resources remains constant (R=10) and different number of tasks have been taken for performance evaluation.

Table-1 represents the resource (VM) configuration in each host.

Table-1: VM Configuration

No. of CPU (Processing Element)	1
Size	10,000 (in MB)
RAM	2048 (in MB)
Bandwidth	1000 (in mbps)

Table-2 represents the each host configuration at Datacenter.

Table-2: Host Configuration

RAM	2048*2(in MB)
Storage	1000000(in MB)
Bandwidth	1000 (in mbps)

The results for makespan performance metric have been represented in a tabular form and graphically as well, where two different numbers of cloudlets/tasks sizes have been taken and the simulation has been done on each task size for 20 times and then the average has been taken.

Table-3 Makespan Analysis (in ms)

No. of cloudlets	Min-Min	Max-Min	HYMM
1000	778.8522	773.18071	760.6267
5000	3953.752	3977.0193	3945.840

Table-3 shows the makespan values of Min-Min, Max-Min and proposed algorithm with two different scenarioes. First, 1000 cloudlets have been taken and then for showing performance on high laod, 5000 cloudlets have been taken.

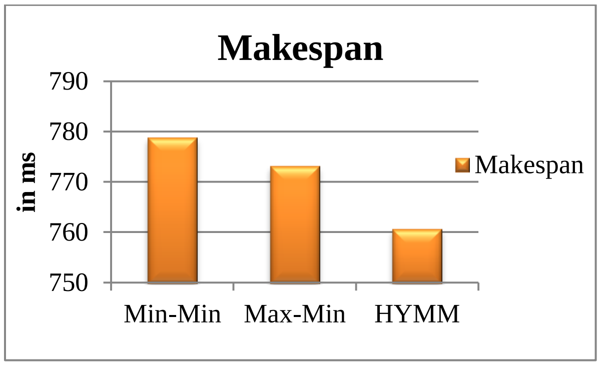

Figure 2: Comparison of Makespan (1000 cloudlets)

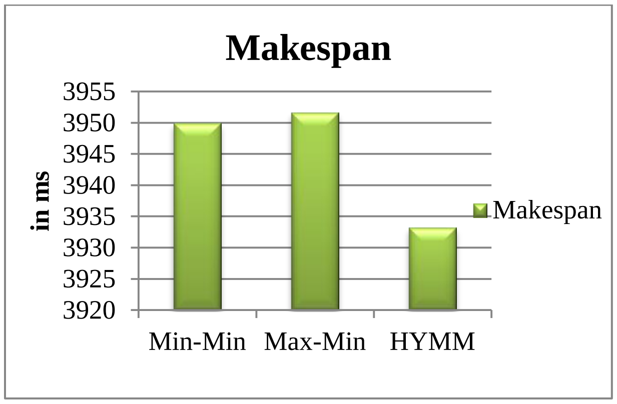

Figure 3: Comparison of Makespan (5000 cloudlets)

Figure 2 and 3 represent the comparison of makespan among the three task scheduling. When number of cloudlets are 1000, Max-Min outperforms Min-Min but HYMM outperforms both algorithms.

Now the results for Average Resource Utilization have been represented in a tabular form and graphically as well in figure 4 and 5.

Table-4: Average Resource Utilization Analysis

No. of cloudlets	Min-Min	Max-Min	HYMM
1000	53.5470	53.8307	54.4464
5000	59.8857	59.8386	60.0687

Table-4 represents the average resource utilization rate of Min-Min, Max-Min and proposed algorithm in two different scenarioes, first with 1000 cloudlets and then for showing performance on high load, 5000 cloudlets have been taken.

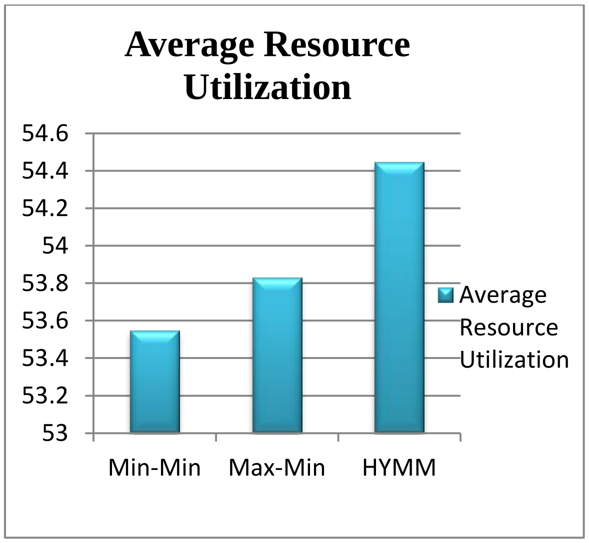

Figure 4: Comparison of Average Resource Utilization (1000 cloudlets)

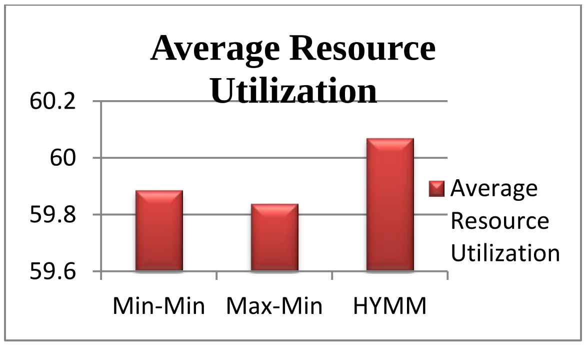

Figure 5: Comparison of Average Resource Utilization (5000 cloudlets)

From figure 4 and 5, it can be concluded that HYMM does better utilization of resources than Min-Min and Max-Min. Here resource utilization rate is between 0 and 1.
The results for load balancing factor have been presented first in table-5 and then figure 6 and 7 shows them graphically.

Table-5: Load Balancing Factor Analysis

No. of cloudlets	Min-Min	Max-Min	HYMM
1000	58.7181	57.9677	58.91924
5000	52.9070	53.0221	53.14008

Table-5 shows the load balancing values of Min-Min, Max-Min and proposed algorithm with in different scenarioes, first with 1000 cloudlets and then for showing performance on high laod, 5000 cloudlets have been taken.

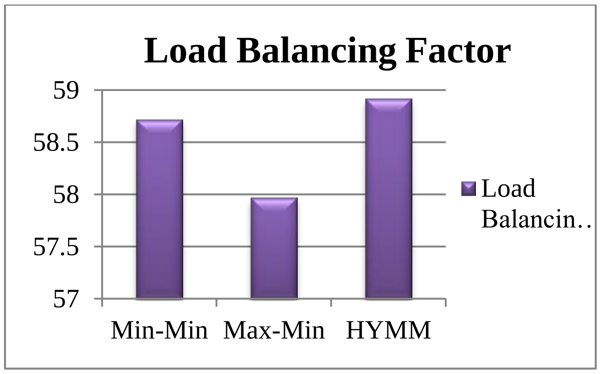

Figure 6: Comparison of Load Balancing Factor (1000 cloudlets)

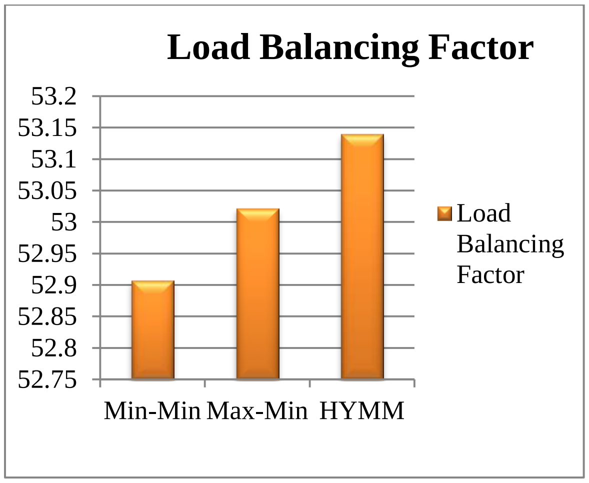

Figure 7: Comparison of Load Balancing Factor (5000 cloudlets)

From above figures, it is concluded that HYMM produces better load balancing than the Min-Min and Max-Min algorithms.

Hence from above results, it can be concluded that HYMM performs better for makespan, average resource utilization and load balancing performance metrics. It produces better results than Min-Min and Max-Min for above discussed parameters.

V. CONCLUSION AND FUTURE WORK

Task Scheduling in cloud computing is a tedious and critical task and for achieving better cloud usage, scheduling is main concern in cloud environment. In this paper, a new and efficient scheduling algorithm has been proposed, which utilizes the advantages of two fundamental and traditional algorithms i.e. Min-Min and Max-Min. It produces better makespan, resource utilization and load balancing. The time complexity of this algorithm is same as the Min-Min and Max-Min algorithm but it performs better results by using their pros. Performance Evaluation of this algorithm has been performed on CloudSim toolkit which provides an efficient simulation environment. In future this algorithm can be enhanced by giving priorities to each task and can be performed on actual cloud.

REFERENCES

[1] T. Kokilavani and Dr. D.I. George Amalarethinam, "Load Balanced Min-Min Algorithm for Static Meta-Task Scheduling in Grid Computing," International Journal of Computer Applications, vol. 20, no. 2, pp. 43-49, April 2011.

[2] Gao Ming and Hao Li, "An Improved Algorithm Based on Max-Min for Cloud Task Scheduling," Recent Advances in Computer Science and Information Engineering in Springer Berlin Heidelberg, vol. 125, pp. 217-223, January 2012.

[3] Neha Sharma, Sanjay Tyagi, and Swati Atri, "A Survey on Heuristic Approach for Task Scheduling in Cloud Computing," International Journal of Advance Research in Computer Science, vol. 8, pp. 1089-1092, March-April 2017.

[4] Huankai Chen, Frank Wang, Na Helian, and Gbola Akanmu, "User-Priority Guided Min-Min Scheduling Algorithm For Load Balancing in Cloud Computing," IEEE, 2013.

[5] Gaurav Sharma and Puneet Banga, "Task Aware Switcher Scheduling for Batch Mode Mapping in Computational Grid Environment," International Journal of Advanced Research in Computer Science and Software Engineering, vol. 3, no. 6, pp. 1292-1299, june 2013.

[6] Kobra Etminani and M. Naghibzadeh, "A Min-Min Max-Min Selective Algorihtm for Grid Task Scheduling," in ICI 2007. 3rd IEEE/IFIP International Conference in Central Asia , 2007.

[7] S.Devipriya and C.Ramesh, "Improved Max-Min Heuristic Model For Task Scheduling In Cloud," in International Conference on Green Computing, Communication and Conservation of Energy (ICGCE), IEEE, 2013, pp. 883-885.

[8] Rajwinder Kaur and Pawan Luthra, "Load Balancing in Cloud System using Max Min and Min Min Algorithm," International Journal of Computer Applications (0975 – 8887)National Conference on Emerging Trends in Computer Technology (NCETCT-2014), pp. 31-34, 2014.

[9] J.Y Maipan-uku, A. Muhammed, A. Abdullah, and M. Hussin, "Max-Average: An Extended Max-Min

Scheduling Algorithm for Grid Computing Environment," Journal of Telecommunication, Electronic and Computer Engineering, vol. 8, pp. 43-47, 2015.

[10] Neha Sharma, Sanjay Tyagi, and Swati Atri, "A Comparative Analysis of Min-Min and Max-Min Algorithms based on the Makespan Analysis," International Journal of Advance Research in Computer Science, vol. 8, pp. 1038-1041, March-April 2017.

[11] Gaurang Patel, Rutvik Mehta, and Upendra Bhoi, "Enhanced Load Balanced Min-Min algorithm for Static Meta Task Scheduling in Cloud Computing," Elsevier, pp. 545-553, 2015.

[12] Christoph Buchheim and Jannis Kurtz, "Min-max-min robustness: a new approach to combinatorial optimization under uncertainty based on multiple solutions," Elsevier, pp. 45-52, 2016.

[13] J. Kok Konjaang, J.Y. Maipan-uku, and Kumangkem Kennedy, "An Efficient Max-Min Resource Allocator and Task Scheduling Algorithm in Cloud Computing Environment," International Journal of Computer Applications, vol. Volume 142, no. No.8, pp. 25-30, May 2016.

[14] O. M. Elzeki, M. Z. Reshad, and M. A. Elsoud, "Improved Max-Min Algorithm in Cloud Computing," International Journal of Computer Applications , vol. 50 , no. 12, pp. 22-27, July 2012.

[15] Nidhi Bansal, Amit Awasthi, and Shruti Bansal, "Task Scheduling Algorithms with Multiple Factor in Cloud Computing Environment," Springer India, pp. 619-627, 2016.

[16] Santhosh B. and Manjaiah D.H., "An Improved Task Scheduling Algorithm based on Max-min for Cloud Computing," International Journal of Innovative Research in Computer and Communication Engineering(IJIRCCE), vol. 2, no. 2, pp. 84-88, May 2014.

[17] D.I. George Amalarethinam and S. Kavitha, "Enhanced Min-Min Algorithm for Meta-task Scheduling in Cloud Computing," IJCTA , pp. 85-91, 2016.

[18] Sameer Singh Chauhan and R. C. Joshi, "A Weighted Mean Time Min-Min Max-Min Selective Scheduling Strategy for Independent Tasks on Grid," published in IEEE, 2nd International Advance Computing Conference, pp. 4-9, 2010.

[19] Rodrigo N. Calheiros, Rajiv Ranjan, Anton Beloglazov, C´esar A. F. De Rose, and Rajkumar Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," Software – Practice and Experience, John Wiley & Sons, pp. 23-50, June 2011.

[20] Rajkumar Buyya, James Broberg, and Andrej Goscinskki, Intoduction to Cloud Computing. New Delhi, India: Wiley India Pvt.Ltd.

[21] Gaurang Pate and Rutvik Mehta, "A Survey on Various Task Scheduling Algorithm in Cloud Computing," International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), vol. 3, no. 3, pp. 715-717, March 2014.