International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)
Web Site: www.ijettcs.org Email: editor@ijettcs.org
Volume 6, Issue 5, September- October 2017 ISSN 2278-6856

Optimized Task Scheduling Algorithm for cloud computing environment

Poonam Rani[1], Pooja Nagpal[2]

[1]Research Scholar (CSE) Rayat Group of Institutions Railmajra,Punjab India.

[2]Associate Professor (CSE) Rayat Group of Institutions Railmajra,Punjab India.

Abstract

Cloud Computing is a rapidly growing technology and it offers online accessible resources to the users. These resources include applications, servers, networking, and storage etc. and provide security, elasticity, scalability and low cost. Most of the businesses are migrating from on-premise to the cloud and millions of users access these services daily via the internet. So it is very important to apply appropriate scheduling technique to process a large amount of data and to do resource utilization more efficiently with better performance. This research paper presents new proposed algorithm which is using benefits of both Enhanced Max-Min and Max-Min algorithms together. In this proposed algorithm, there are two sets of resources created by considering their MIPS speed where the first one contains the resources with maximum execution time and the second one contains the resources with minimum execution time. The large task is assigned to the resource if the available resource is from the second set and average length task is assigned to the resource if the available resource is from the first set. The simulation tool used for testing is WorkflowSim. Test results display that the new proposed algorithm represents enhanced resource utilization with better makespan.

Keywords :- Cloud Computing, Load Balancing, Makespan, Energy Consumption, FCFS, Max-Min.

1. INTRODUCTION

Cloud computing is a new ordinary technology and becoming very famous among businesses, companies, institutions and industries. Cloud computing provides various on- demand IT resources which are accessible via internet and follows pay as you go model. That means users can access these resources whenever they want to use and pay forit as per usage. There is no need to pay for the resources which they are not using. There are some advantages of cloud computing over the traditional computing technique. It provides agility; speed and flexibility that means users can scale up and scale down the service capacity as per the requirements. It also offers scalability that means users can add resources in case of increasing load and remove resources in case of decreasing load. There are main five characteristics of cloud computing such as on-demand self-service-user can access the required services by signing up without waiting so much, Resource pooling - resources are shared among several users, Measured service use of resources is measured and billing of the usage is delivered, Broad network access users can access the resources through usual platforms like laptop, mobile phone etc., Rapid elasticity - user can scale the resource capacity to fulfill the increasing demands.

So the number of data stored and handled in a cloud is very huge and this data amount is increasing day by day. Proper task scheduling technique is important to manage the large bunch of tasks that arrived in cloud concurrently. There is a need for better resource utilization to handle the large amount of load. Assignment of tasks to the resources is the task scheduling. It is necessary to allocate all the resources which are available for processing tasks instead of using few resources. It reduces the burden of resources while processing tasks. Better scheduling technique is also useful for cloud providers to get more revenue. In a cloud, task scheduling and resource organization play an important role in effective resource management [21].There is not one uniform way of scheduling in cloud computing. Various task scheduling techniques already exist.

But this research work presents proposed task scheduling algorithm reduces the total length of scheduling and utilizes resources properly.

2. RELATEDWORK

In this section we discuss relevant related research divided in the topics such as Problem statement, Task scheduling approaches, Cloud simulator.

a. Problem Statement:

Task scheduling is a process in which tasks that need to be executed are assigned to the available resources in the cloud through the internet. If a provision of resources is not done appropriately, most of the cloud services remain unused. To fulfill the user demands and application requirements, the service provider uses Resource Allocation Policy which is the combination of all the actions that has been taken to allocate scarce resources. For this cloud service provider need to have an idea about how much and which type of resources are required to execute a particular task. Before provisioning resources, a service provider should know about all available resources and also the status of each resource. Whereas user should know about all needs of the application.This is useful to escape from situations such as scarcity, fragmentation, and contention of resources as well as to avoid under and over provisioning. The problem will occur if more than one task makes effort to use the same resource simultaneously, limited set of resources are available as compared to user requirements or task is allocated to extra resources instead of assigning to exact one. There is a necessity of proper resource management because users resource request is unpredictable for service provider and users are looking for the services which take less time and money to complete the task.

There are two types of resources in cloud environment i.e. virtual and physical. Figure 1 shows the mapping of virtual to physical resources. Virtualization technology is used for assigning users task to the resources. These resources are distributed as per the user demand. Discovering efficient resource utilization technique is quite difficult in cloud computing[25].Task scheduling is a critical process because cloud service provider has to find a suitable way for scheduling the tasks arrived from users to the available resources within a given cloud environment. The scheduling algorithm should be useful in increasing performance with a decrease in total execution time and completion time of tasks.

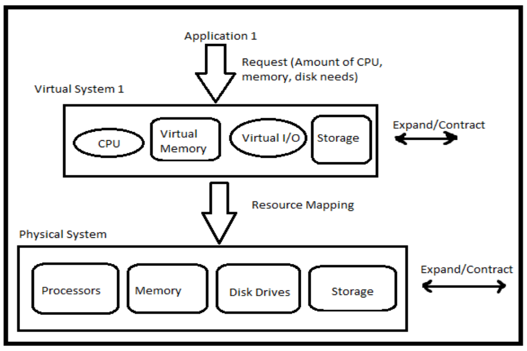

Figure 1: Mapping of virtual to physical resources

There is a need for appropriate scheduling algorithm in grid and cluster computing because a large amount of data and computing processing has been done within that distributed environment. Currently, various scheduling algorithms are in use which is mostly used to make task completion time shorter [8].Hence the effective process of scheduling a task has huge importance in the cloud computing. Task scheduling plays an important role to get more profit by proficiently making growth in the processing of cloud environment.

Task scheduling algorithm is used to utilize resources more efficiently while reducing the overall execution time of tasks and to distribute the total load on various computing resources. Task scheduling also plays the main role in the development of consistent and elastic systems. Scheduling technique is used to map tasks to the flexible resources in compliance with resilient period and to discover the suitable execution order of tasks. Dynamic scheduling and static scheduling algorithms are the two key types of algorithms that are based on job scheduling. Each algorithm has some limitations as well as some advantages. The static scheduling algorithm has less overhead than dynamic scheduling algorithm whereas the performance of dynamic scheduling algorithm is much better as compared to the static scheduling algorithm.

In cloud computing, there are two types of scheduling algorithms. The first type is online mode heuristic scheduling algorithms and the second type is batch mode heuristic algorithms. Tasks are mapped to the resources immediately after the ar- rival of tasks in online mode heuristic scheduling. An example of this is the most fit task scheduling. First Come First Serve (FCFS), Round robin, Max-Min are the examples of batch mode heuristic algorithms. It initializes the scheduling of tasks after a particular time period and it put all incoming tasks into the queue. In the cloud, scheduling is done in three steps which are discovery and filtering of resource, selection of resource and mapping of the task. Figure 2 shows the scheduling process in cloud. Resources available in the system are discovered by Datacenter Broker and it also gathers the related information such as a status of the resource, etc. The second step is the selection of resource among all available resources and this selection is takes place by considering definite parameters of both resource and task. The last step is to map the task to the selected resource [20].

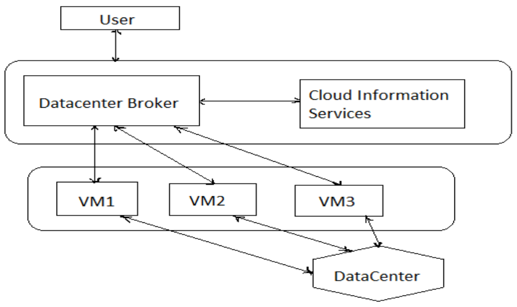

Figure 2: Scheduling Process in Cloud

b. Task Scheduling Approaches:

We start our study with traditional algorithm First Come First Serve (FCFS). This scheduling algorithm schedule the tasks on a first come first serve basis. This is the fast and simplest algorithm. Round Robin algorithm uses the time-sharing technique while scheduling the tasks. The particular time slice is assigned to each task. This time slot is nothing but the CPU time. If selected task became unable to complete its execution within given CPU time, then next task in the ready queue is assigned to pre- empted CPU and a previous uncompleted task is pushed at the end of the ready queue. Small tasks took more time for execution in Round Robin algorithm because they share the CPU time with large tasks instead of the complete execution quickly[20]. Minimum execution time (MET) algorithm is mostly used to reduce the total execution time of each task within a heterogeneous environment. Mapping of application to resources is NP problem .For efficient resource utilization, it is also necessary to know in which order task would be executing. MET algorithm uses first come first serve technique while assigning resources [12]. MET select a resource which will execute the arrived task in less time as compared to all other resources. But it does not consider whether the selected resource is currently available to perform a task or not [9]. This is increasing a load on cloud services. Task has to wait until the assigned resource becomes free.

Minimum Completion Time (MCT) algorithm is used for assigning submitted application to the resource which is able to complete the particular task with less time. In short, MCT selects the resource having minimum completion time but this algorithm works on tasks one by one [15]. When a task is submitted by the user, opportunistic load balancing (OLB) algorithm find out the available resource and assigns a task to it. If no resource is available at a current time, then OLB assign a task to the resource which will become free first and if there are a group of available resources, OLB select any resource randomly to assign a task. Hence there is a possibility of scheduling task to the resource which takes more time for execution and completion [14]. Min-Min algorithm is used to perform tasks first which takes less time for completion. Therefore, huge tasks stay in queue until all small tasks finished their execution [20]. Initially, all the tasks are not mapped to the resources. Consider that there are three tasks, Min-Min algorithm first calculates the completion time of each task on each resource.

Max-Min algorithm is used to perform tasks first which takes more time for completion. Therefore, small tasks stay in queue until all huge tasks finished their execution[20]. Similar to Min-Min algorithm, all the tasks are not scheduled at first. A resource which takes minimum time for task completion is chosen. Then the task with large completion time is mapped to the selected resource. All the large tasks are executed first. Therefore, large tasks do not have to wait for a long time for execution. Mapping of tasks to resource continues until all tasks executed. Like Min-Min, available time of the machine is also modified. For example, if the first task requires 80 secondsfor completion, then execution time of remaining tasks are increased by 80 seconds [21].Resource Away Scheduling Algorithm (RASA) provides benefits of both Min-Min and Max-Min algorithm. RASA is used to schedule a task based on either Min-Min or Max-Min according to the current situation. Selecting small task first and mapping it to the fast resource is the work of Min-Min algorithm. This type of scheduling is useful when there are less number of small tasks than large tasks. If submitted task contains only one large task and all other are small tasks, then using Min-Min algorithm will increase total make-span because the large task executed at last after completion of all tasks. In such situations, using Max-Min algorithm is a beneficiary. This algorithm allows the large task to be executed first and during execution of large tasks, set of small tasks can be executed simultaneously. Executing large tasks first may result in high response time.

RASA algorithm provides advantages of choosing suitable algorithm either Min-Min or Max-Min alternatively at any stage of scheduling. For example, if there are all small tasks initially then RASA prefer Min-Min algorithm for scheduling task to a resource. But if on every large task arrived suddenly, then RASA will change the preference to Max-Min algorithm for task completion. Some tests have proven that better scheduling happens when using Max-Min technique if there are an even number of available resources initially otherwise using Min-Min technique. This algorithm increases efficiency and produces better response time [16]. Heterogeneous Earliest Finish Time algorithm is popular for giving a better result within a less time. It is a static scheduling algorithm. HEFT heuristic is more useful among all other heuristics because of it provides good scheduling within a heterogeneous environment with less time. HEFT works in two stages. In the first stage, HEFT set priority for all submitted tasks and in the second stage, choose the task with the highest priority and assign it to the resource which finishes execution process early. After completion of the first task, this process is repeated for all the remaining tasks. Each task is assigned to the best resource which results in earliest finish time [24].For setting the priority of each task, HEFT first computes how much execution time will require for each submitted task. It also computes time required to transfer result between the resources if there are some tasks which are dependent on each other for completion. HEFT offers great performance and speed [3].

As explained earlier that Max-Min algorithm first chooses a resource which takes less time for task completion and then long task i.e. a task takes maximum time for execution is selected and assigned to that resource. But, in improved Max-Min algorithm, a task having large completion time is mapped to their source which will execute the assigned task with less time.

There are main two types of resource management system, one is local resource management system where an ordinary interface is available to use distant resources and second is global resource management system where local resource management systems are organized with in virtualized organization to access and handle resources [22]. Resource allocation strategy is the combination of all actions of service providers to fulfill the requirements of cloud application by assigning rare resources. It is important to identify which type of and how many resources are required to execute the task. To get ideal resource allocation strategy, it is necessary to provide resource assignment sequence and time as an input.

Best resource allocation strategy is vital to avoid under provisioning, over provisioning, resource contention, resource fragmentation and scarcity of the resource. In the case of under-provisioning, only a few resources are allocated to the application than it actually required. Under-provisioning happens due to the assignment of resources by the cloud service providers. In a case of over provisioning, there is a set of extra resources available than the application exactly required for completion. Over provisioning happens due to the guess of users about total requirements of resources to fulfill the task within estimated time period. Resource contention means that conflict occurs while accessing shared resources like storage, memory. This problem happens when multiple users want to access shared resource. It causes low system performance. In Resource fragmentation, a user is not able to get access to resources although there are sufficient resources available because resources are fragmented into the small objects. So that it is not possible to utilize such resources ideally. The scarcity of Resource means that demand for resources is so high as compared to the actual available resources. In this case, a user can not access the required resources. Therefore, resource allocation strategy should get input from the user as well as from the service provider to avoid above stated inconsistencies. Load balancing is done at two stages in cloud computing, first one is while assigning applications to the physical computers for balancing a load on it and the second one is while assigning multiple requests to the application instances for balancing a load on applications. Although resources offered by the cloud are reliable but it causes some difficulties during resource management and allocation. From the service provider's point of view, it is unfeasible to guess the demands of application and demands of users. Users want to finish a task with less time and cost. Therefore, cloud requires effective resource allocation method [1].

In task scheduling, the task first comes to cloud coordinator. Cloud coordinator sends this task to the data center. There are lots of hosts in a data center which contains many virtual machines. It is possible to organize and remove the host according to demand [26].In the process of resource allocation, virtual machine manager forward needs of a task to the service provider and request for resources as per requirements. Service provider asks resource owner whether the requested resources are available or not. Then provider gets approval of accessing available resources from resource owner. The service provider also provisions resources to create virtual machines. The virtual machine is nothing but a virtualized server [6].

c. Cloud Simulator

WorkflowSim is the toolkit used for simulation of scheduling algorithms. It is the advanced version of CloudSim by offering better workflow management and accurate evaluation. CloudSim [5] is used for simulation of cloud services and infrastructure. CloudSim allows executing only single workload. It does not consider failures and overheads. It does not support clustering and job dependencies. Unlike other simulators, failures and overheads occurred in the heterogeneous system are considered into the WorkflowSim. It also supports clustering. Figure 3 shows the WorkflowSim Architecture. WorkflowSim contains multiple layers such as Failure monitor, Failure generator, Clustering engine, Workflow engine and Workflow mapper along with the Workflow scheduler which is present into the CloudSim. Workflow mapper has used for mapping non-concrete workflows to the actual workflows which are reliant on execution sites. Data dependencies are managed by Workflow engine. Tasks are scheduled to the resources with the help of Workflow Scheduler. Small jobs are combined into a large one by using Clustering engine [7].

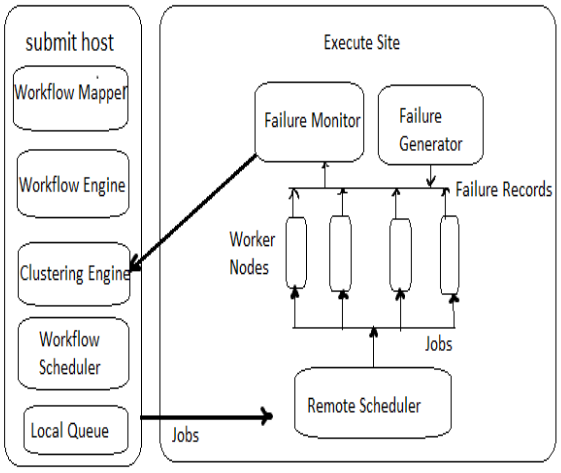

Figure 3: WorkfowSim Architecture

3. PROPOSED METHODOLOGY

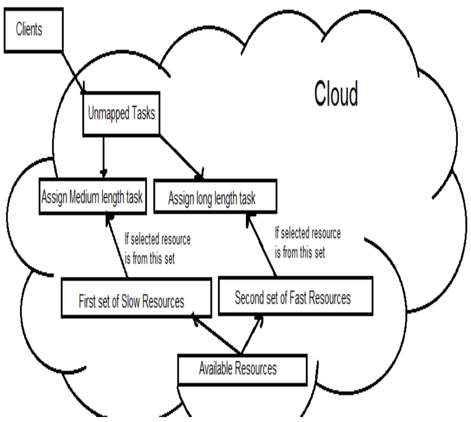
Clients
Cloud
Unmapped Tasks
Assign Medium length task
Assign long length task
If selected resource is from this set
If selected resource is from this set
First set of Slow Resources
Second set of Fast Resources
Available Resources

Figure 4: System Architecture

Our proposed approach divides all available resources into two sets where first set is of the slow resources and second set is of the fast resources. This partitioning is done according to the MIPS speed of each resource. Suppose that there are 10 available re- sources. The first step is to sort these:

3.1 Simulation Tool

This research project uses WorkflowSim. It is very popular for simulation purpose. It is the enhanced version of CloudSim. There are many simulation tools are available such as GridSim, CloudSim, WorkflowSim. But we chose Workflow Sim as a cloud simulator because it offers workflow level support and clustering. But GridSim and CloudSim do not support clustering and also do not consider overheads. WorkflowSim is a simulation framework to test and validate the accuracy of the proposed algorithm [7].WorkflowSim is open source toolkit written in Java.

4. DESIGN SPECIFICATION

4.1 Pseudo Code

The proposed algorithm is the combination of Enhanced Max-Min and Max-Min strategies. When using these strategies separately, it generates problem like delay in large task completion, response time issues. But if we use these strategies combine, it will prove beneficiary in terms of performance and makespan.

4.2 Workflow Diagram

Workflow diagram is basically used to symbolize overall process into steps. It depicts how the algorithm works. This flow chart represents all activities such as sorting, dividing, and selection of resource, task assignment criteria etc. These steps are useful for the easy understanding of concept or algorithm. Client or user send tasks for execution. Initially, all tasks are unmapped. Then sorting resources and partitioning them into two sets. Find out the resource having minimum execution time. If this resource found out in the first set, assign medium size task to it else assign the large task. After completion of task execution, release that resource and make it available for other tasks. The same process repeats until all tasks finished.

Algorithm1: Proposed algorithm

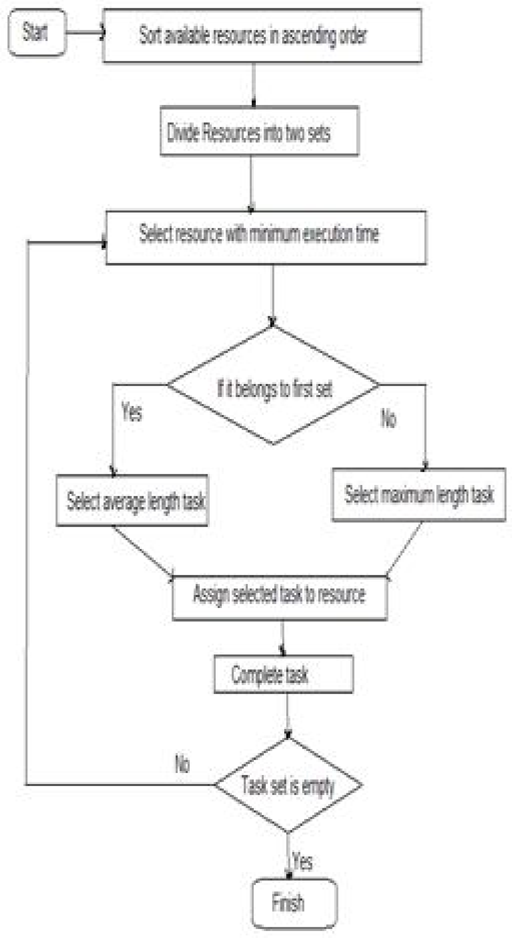
Start
Sort available resources in ascending order
Divide Resources into two sets
Select resource with minimum execution time
If it belongs to first set
Yes
No
Select average length task
Select maximum length task
Assign selected task to resource
Complete task
No
Task set is empty
Yes
Finish

Figure 5: Flowchart of algorithm

5.METHODOLOGY

```
For all tasks Ti in MT
Organize ascending order of N resources according to MIPS speed
Create two sets of resources by dividing N as N/2
While there are tasks in MT
Find the available resource Ri with minimum execution time
If selected Ri is from the first set
Select the medium length task among all tasks
Else
Select the maximum length task among all tasks
Assign selected task Ti to the Resource Ri
Delete assigned Ti from the MT
End While
```

We have studied various task scheduling algorithms in literature review section. Each algorithm has its own benefits and limitations. Some algorithms are implemented to overcome some limitations of the previously existing algorithm. For example, the limitation of Min-Min algorithm is large task may have to wait for execution until all task gets finished. But Max-Min algorithm is implemented to overcome such problem to get better performance than Min-Min algorithm. Max-Min algorithm executes long tasks first, so that it may effect on total response time. Resource Aware Scheduling Algorithm is another approach which allow the system to use both Min-Min and Max-Min techniques alternatively so that to get proper load balancing and enhanced response time. Enhanced Max-Min algorithm is useful in a situation like if there is a very long task among all ar- rived tasks, then giving a resource to it first will increase a make-span i.e. total length of scheduling. So the solution offered by Enhanced Max-min is to choose a task which is average in length and assigning it to the fast resource. Whereas Heterogeneous Earliest Finish Time algorithm assigns highest priority task to the resource which will finish it in less time. But all above algorithms still faces the one problem when a long task may have mapped to the resource which takes much time for execution results in increasing length of scheduling i.e. Makespan. To overcome this problem we propose an algorithm which reduces the makespan.

6. IMPLEMENTATION

The software used for this experiment is Eclipse Java Neon and WorkflowSim1.0. The configuration of the system used for this experiment are 64-bit Windows 10 operating system, Intel(R) Core(TM) i5 CPU 2.20 GHz, 8GB RAM. We have developed the proposed algorithm in Java language. Then Workflow Sim simulator is used to test the developed algorithm. It offers workflow level support during the simulation

To test the proposed algorithm, one data center, one scheduler, and 5 virtual machines are used first.

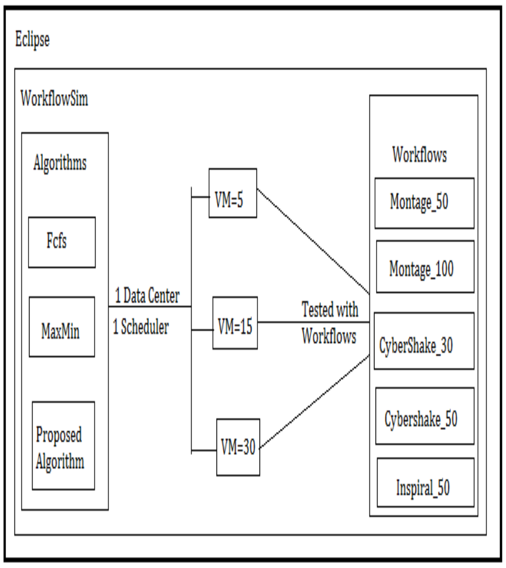

Figure 6: High-level Overview of Experimental Setup

Then Proposed algorithm tested by using five workflows named Maontage50,Montage100, CyberShake30, CyberShake50,and Inspiral50. The reason behind choosing these work- flows for testing is that previous research papers [11][13]have used these datasets to test MaxMin algorithm. The purpose of this paper is to show that performance of proposed algorithm is better than the MaxMin algorithm.

Then the numbers of virtual machines are increased by 15 and 30 respectively to test the algorithm on the same workflows as mentioned above. There is one DAX file in WorkflowSim which contains all the workflows. All these workflow files are in .xml format. There are a maximum number of large tasks in Montage workflow. Montage50 contains total 50 tasks and Montage100 contains total 100 tasks. There are less number of large tasks in Inspiral workflow. Inspiral50 contains total 50 tasks. Large sized tasks in CyberShake are in average numbers. CyberShake30 means it contains 30 tasks and CyberShake50 means it contains 50 tasks [4].

We have also developed FCFS and MaxMin algorithms in Java using their pseudo codes for the simulation purpose. Because we compare our proposed algorithm with FCFS and MaxMin to check whose performance is better.Figure 6 shows the high-level overview of the Experimental setup.

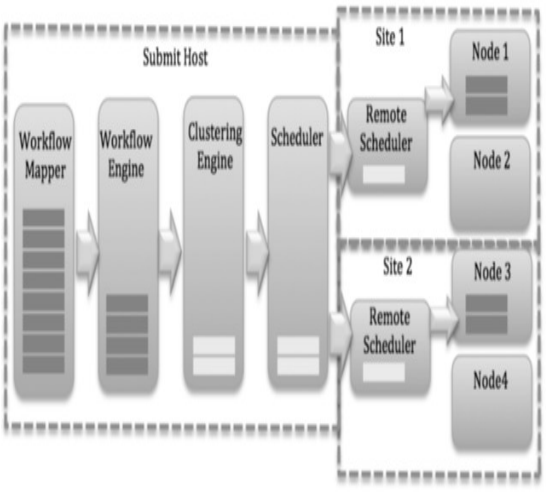

Figure7: Interaction between WorkflowSim Components

We use several Workflow Sim components such as Workflow Mapper, Clustering Engine, Workflow Engine, and Workflow Scheduler. They plays an important role during execution of each workflow. Figure 7 shows their interaction. Next, we provide description per each component. Workflow Mapper: The job of Workflow Mapper is to import workflow file which is in the format of XML. Listing of tasks and then mapping are done by Workflow Mapper.

Clustering Engine: The overhead of scheduling is minimized with the help of Clustering Engine by merging tasks into the jobs. A job is a unit recognized by execution site and it includes a number of tasks that are performed in parallel or in the sequence that means horizontal clustering or vertical clustering.

Workflow Engine: Jobs created during clustering are handled and managed by Work- flow Engine. Workflow Engine forward job to the scheduler only if all the previously released jobs have executed well.

Workflow Scheduler: Scheduling of jobs to the worker nodes is done by Workflow Scheduler. Worker node should be ideal for the process of scheduling jobs to it in the remote scheduler.

These components are coordinated with each other by managing their own message queue. That means for every iteration, Clustering Engine review if it needs to forward a job to the scheduler or if Workflow Engine has sent tasks to it. The simulation got over if message queue of all the components is empty [7].

7.EVALUATION

a. Makespan Evaluation for 5 Virtual Machines:

Table 1: Makespan Comparison of Scheduling Algorithms where VM=5

Algorithms / Workflows	FCFS Algorithm (in milliseconds)	Max-Min Algorithm (in milliseconds)	Proposed Algorithm (in milliseconds)
Montage_50	302.02	235.22	220.41
Montage_100	598.16	481.49	459.64
CyberShake_ 30	716.08	518.45	421.4
CyberShake_ 50	717.92	582.52	550.16
Inspiral_50	5150.53	4377.31	4128.34

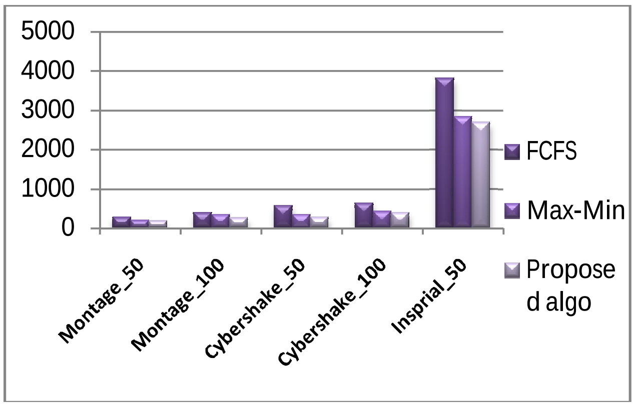

Fig 8: Makespan Comparison of Scheduling Algorithms where VM=5

b. Makespan Evaluation for 15 Virtual Machines:

Table 2: Makespan Comparison of Scheduling Algorithms where VM=15

Algorithms / Workflows	FCFS Algorithm (in milliseconds)	Max-Min Algorithm (in milliseconds)	Proposed Algorithm (in milliseconds)
Montage_50	277.90	183.62	167.27
Montage_100	394.99	315.46	263.13
CyberShake_ 30	562.50	331.88	276.42
CyberShake_ 50	627.08	422.56	391.88
Inspiral_50	3812.37	2829.54	2685.83

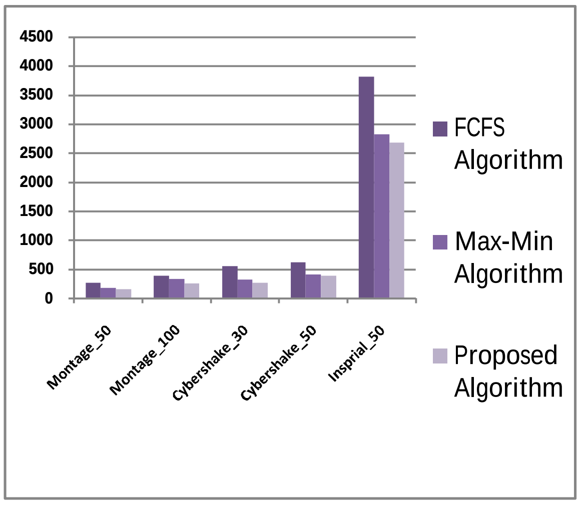

Fig 9: Makespan Comparison of Scheduling Algorithms where VM=15

C. Makespan Evaluation for 30 Virtual Machines:

Table 3: Makespan Comparison of Scheduling

Algorithms / Workflows	FCFS Algorithm (in milliseconds)	Max-Min Algorithm (in milliseconds)	Proposed Algorithm (in milliseconds)
Montage_50	186.9	124.74	110.12
Montage_100	234.15	203.51	175.27
CyberShake_ 30	425.09	276.66	257.37
CyberShake_ 50	485.17	314.35	290.29
Inspiral_50	3206.05	1714.02	1679

Algorithms where VM=30

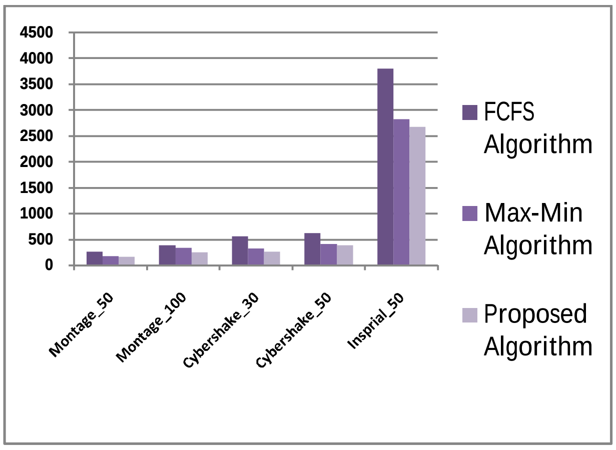

Fig 10: Makespan Comparison of Scheduling Algorithms where VM=30

8. DISCUSSION

Results of above evaluations show that proposed algorithm completes tasks execution with lower makespan and higher performance as compared to scheduling algorithms MaxMin and FCFS. Performance of proposed algorithm is around 8 percentages better than MaxMin algorithm and around 27 percentages better than FCFS for 5 virtual machines. Performance of proposed algorithm is around 10 percentages better than MaxMin algorithm and around 38 percentages better than FCFS for 15 virtual machines. Performance of proposed algorithm is around 10 percentage better than MaxMin algorithm and around 39 percentage better than FCFS for 30 virtual machines. Results shows that proposed algorithm behaves better in terms of makespan after testing it with Montage50, Montage100, CyberShake30, Cybershake50 and Inspiral50 workflows. As we increased a number of virtual machines which are used for simulation, still the total length of scheduling for the proposed algorithm is less than the MaxMin and FCFS.

9. CONCLUSION AND FUTUREWORK

9.1 CONCLUSION

This research work presents new task scheduling algorithm based on resource partitioning to minimize the total length of scheduling and proper resource utilization. This algorithm is implemented using WorkflowSim simulation tool. Large task is assigned to the fastest resource in case of MaxMin algorithm. There is a limitation of MaxMin which is sometimes large task is mapped to the slow resource. This increases the total length of scheduling. But in proposed algorithm possibility of scheduling long length tasks to the slow resource is reduced. Solution used in this paper is division of resources into two groups according to MIPS speed. If the fastest available resource is from the second group which is the group of fast resources, then the largest task is scheduled to it. If the fastest available resource is from the first group which is the group of slow resources, then the average length task is mapped to it.Results shows that proposed algorithm done efficient resource utilization and has better makespan than existing scheduling algorithms MaxMin and FCFS.

Overall result shows that the proposed algorithm performs better than MaxMin algorithm when it comes to makespan. Proposed algorithm improves the performance of scheduling by 10 percentages as compared to MaxMin. Makespan of proposed algorithm for all workflows such as Montage50, Montage100, CyberShake30, CyberShake50, and Inspiral50 is less than makespan of MaxMin.

9.2. Future Scope

The proposed approach helps us to give the better result with respect to FCFS, MaxMin and MinMin workflow scheduling algorithm to minimize the cost and time of execution while meeting the deadline and budget constraints specified by the user. In future we can develop a new algorithm that minimizes the execution cost and execution time and give better results than proposed algorithm. Changes can be made in some parameters of the proposed algorithm.

REFERENCES

[1] Babu, K. D. and Kumar, D. G. (2012). Allocation strategies of virtual resources in cloud computing networks, Journal of Engineering Research and Applications 201: 51–55.
[2] Bhoi, U. and Ramanuj, P. N. (2013). Enhanced max-min task scheduling algorithm in cloud computing, International Journal of Application or Innovation in Engineering and Management 2(4): 259–64.
[3] Bittencourt, L. F., Sakellariou, R. and Madeira, E. R. (2010). Dag scheduling using a look ahead variant of the heterogeneous earliest finish time algorithm, 2010 18th Euromicro Conference on Parallel, Distributed and Network-based Processing,IEEE, pp.27–34.
[4] Brar, S. S. and Rao, S. (2015). Optimizing workflow scheduling using max-min algorithm in cloud environment, International Journal of Computer Applications 124(4).
[5] A. and Buyya, R. (2011). Cloudsim: a tool kit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, Software: Practice and Experience 41(1):23–50.
[6] Cao, Q., Wei, Z.-B. and Gong, W.-M. (2009). An optimized algorithm for task scheduling based on activity based costing in cloud computing, 2009 3rd International Conference on Bioinformatics and Biomedical Engineering, IEEE, pp. 1–3.
[7] Chen, W. and Deelman, E. (2012). Workflow sim: A toolkit for simulating scientific work- flows in distributed environments, E-Science (e-Science), 2012 IEEE 8thInternational Conference on, IEEE, pp.1–8.
[8] Elzeki, O., Reshad, M. and Elsoud, M. (2012). Improved max-min algorithm in cloud computing, International Journal of Computer Applications 50(12).

[9] Etminani, K., Naghibzadeh, M. and Yanehsari, N. R. (2009). A hybrid min-min max-min algorithm withimproved performance, Department of Computer Engineering, Univer- sity of Mashad.

[10] Gouda, K., Patro, A., Dwivedi, D. and Bhat, N. (2014). Virtualization approaches in cloud computing, International Journal of Computer Trends and Technology (IJCTT).

[11] Haladu, M. and Samual, J. (n.d.). Optimizing task scheduling and resource allocation in cloud data centerusing enhanced min-min algorithm, IOSR Journals (IOSR Journal of Computer Engineering) 1(18):18–25.

[12] Hemamalini, M. and Srinath, M. (2015). Memory constrained load shared minimum execution time grid task scheduling algorithm in a heterogeneous environment, Indian Journal of Science and Technology 8(15).

[13] Konjaang, J. K., Maipan-uku, J. and Kubuga, K. K. (2016). An efficient max-min resource allocator and task scheduling algorithm in cloud computing environment, arXivpreprint arXiv:1611.08864.

[14] Maheswaran, M., Ali, S., Siegel, H. J., Hensgen, D. and Freund, R. F. (1999). Dynamic mapping of a class of independent tasks to heterogeneous computing systems,Journal of parallel and distributed computing59(2):107–131.

[15] Moharana, S. S., Ramesh, R. D. and Powar, D. (2013). Analysis of load balancers in cloud computing, International Journal of Computer Science and Engineering 2(2): 101–108.

[16] Parsa, S. and Entezari-Maleki, R. (2009). Rasa: A new task scheduling algorithm in grid environment, World Applied sciences journal7: 152–160.

[17] Priyadarsini, R. J. and Arockiam, L. (2014). Performance evaluation of min-min and max-min algorithms for job scheduling in federated cloud, International Journal of ComputerApplications(0975–8887)Volume.

[18] RajwinderKaur, P. L. (2014). Load balancing in cloud system using MaxMin and MinMin algorithm, National Conference on Emerging Trends in Computer Technology (NCETCT-2014) pp.31–34.

[19] Sagar, M. S., Singh, B. and Ahmad, W. (2013). Study on cloud computing resource allocation strategies, International Journal 1(3): 107–114.

[20] Salot, P. (2013). A survey of various scheduling algorithm in cloud computing environment, International Journal of research and engineering Technology (IJRET), ISSN pp.2319–1163.

[21] International Journal of research and engineering Technology (IJRET), ISSN pp.2319–1163.

[22] Sindhu, S. (2015). Task scheduling in cloud computing, International Journal of Advanced Research in Computer Engineering Technology 4.

[23] Sviji, K. M. A. (2016). Resource managment system in cloud environment: An over- view, International Journal of Advanced Research in Biology, Engineering, Science andTechnology2:357–362.

[24] Tilak, S. and Patil, D. (2012). A survey of various scheduling algorithms in cloud environment, International Journal of Engineering Inventions 1(2): 36–39.

[25] Topcuoglu, H., Hariri, S. and Wu, M.-y. (2002). Performance-effective and low-complexity task scheduling for heterogeneous computing, IEEE transactions on parallel and dis- tributed systems 13(3): 260–274.

[26] Vinothina, V., Sridaran, R. and Ganapathi, P. (2012). A survey on resource allocation strategies in cloud computing, International Journal of Advanced Computer Science and Applications 3(6): 97–104.

[27] Warneke, D. and Kao, O. (2011). Exploiting dynamic resource allocation for efficient parallel data processing in the cloud, IEEE transactions on parallel and distributed systems 22(6): 985–997.