International Journal of Technical Research and Applications e-ISSN: 2320-8163,
www.ijtra.com Volume-2, Special Issue 2 (July-Aug 2014), PP. 01-04

MJHP - JOB SCHEDULING ALGORITHM FOR CLOUD ENVIRONMENT

S.Rekha, R.Santhosh Kumar
Department of Information Technology
Sri Venkateswara College of Engineering
Chennai, India

***Abstract*- Cloud computing can be defined as a type of Internet-based computing, where different services such as servers, storage and applications are delivered to an organization's computers and devices through the Internet. It is a cluster of network where the nodes interact to accomplish a big computational task. Due to its efficiency in running multiple programs simultaneously, cloud computing is emerging as a popular domain in technology. Intense research has thrown light on how the resources are shared and jobs are scheduled amongst the nodes. A proper job-scheduling algorithm is required for the efficient functioning of the cloud environment. The proposed priority based scheduling algorithm for cloud computing is based on factors that govern the functioning of a job.**

***Keywords*- Cloud Computing, Job Scheduling, Priority, Computational Complexity and Level of Parallelism.**

I. Introduction

With the rapid evolving technology, more business organizations are adapting to cloud computing[3] because of its merits such as the ability to process large amount of data and perform complex computations. Thus for a cloud environment to work effectively, proper scheduling has to be designed and implemented as job delays or data loss can prove to be a major hazard for the organization. A cost effective and maximum performance achievable algorithm will prove to be the backbone of the cloud environment. The existing Batch mode heuristic scheduling algorithms (BMHA) are: First Come First Served scheduling algorithm (FCFS), Shortest Job Fastest Resource (SJFR), Longest Job Fastest Resource (LJFR), Min–Min algorithm and Max–Min algorithm. These algorithms consider all jobs with equal importance. The existing Priority Based scheduling Algorithm[1] gives first preference to a job that has highest computational complexity and level of parallelism. However, this algorithm faces the drawback of time wasted in waiting for the high priority job to be executed completely. This time delay can be avoided if medium[2] level job is executed prior to the high priority job. This is the main focus of the proposed MJHP - Medium Job High Priority job scheduling algorithm[7]. Thus, an ideal scheduling[6] algorithm can be developed that satisfies the time constraint as well as achieves best performance. The paper is organized as follows: section 2 deals with the cloud architecture, section 3 describes the proposed algorithm, section 4 provides a picture of performance analysis and section 5 concludes the paper.

II. Cloud Architecture

The cloud architecture comprises of the cloud components that communicative with each other and delivers the output. The frontend platforms visible to the user can be a mobile device or web and back end platform components are servers, storage and a network. This is shown in fig 1.

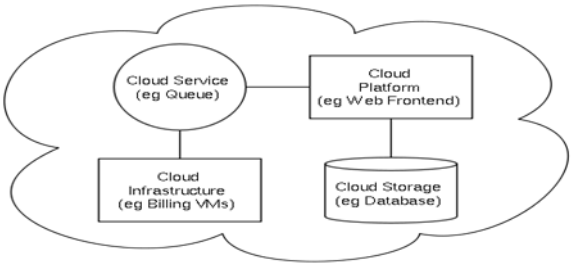

Fig 1. Cloud Computing Architecture

Jobs arrive at the scheduler to be executed along with request for cloud resources[3]. The function of the scheduler is to select how several incoming jobs have to be processed and allocate resources wisely. The overall performance and throughput of the system depends on how the scheduler works. The scenario is depicted in fig 2.

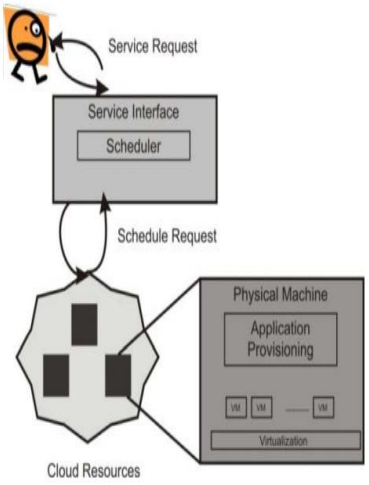

Fig 2 : Cloud job scheduler.

III. Proposed algorithm - MJHP Scheduling in Cloud Computing

Jobs arriving at the scheduler are classified as high, medium and low based on the computational complexity of the job and level of parallelism of the resources. The level of parallelism of a resource and computational power of a job is decided by considering the job parallelism, resource parallelism and job's computational complexity respectively. In the exisiting algorithm, higher priority was assigned to job of higher computational complexity and the resource exhibiting higher level of parallelism .The fastest resource available was assigned to the job of high priority. This method of allocation of resources to the job with high priority[9] resulted in significant waste of time as the processor has to wait till the complex job has to be executed. This threat is overcome, in the proposed MJHP algorithm where preference is given to a job with medium

[2]computational complexity. This priority algorithm optimizes the computational speed of the cloud[4] and reduces the usage of nodes and also shows a consistent performance during execution of the assigned jobs.

A. Computational Complexity

The Directed Acyclic Graph (DAG) gives a visual representation of the task partitioning [10]algorithm which efficiently divides a job into subtasks of appropriate grain size. This graph is shown in fig 3. Each node in DAG represents sequence of operations. Each task[10] can be executed on a processor and the directed arc depicts the transfer of relevant data from one processor to another. The amount of computations involved in a particular node is represented by node weight.

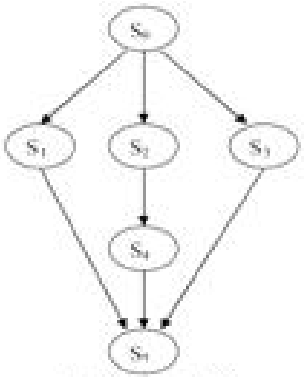

Fig 3 : Directed Acyclic Graph (DAG)

This graph needs to be traversed to find out the longest path. The total sum of the amount of computations involved in each node through which the traversal has been performed leads to computational complexity of the application.

B. Level Of Parallelism

Generally, the amount of parallelism exhibited by a job is computed and analyzed by analyzing it's layered DAG representation. The width of the DAG is equal to the number of sub tasks, which are executed through parallelism. The maximum number of independent instructions getting executed in a unit time (in one clock cycle) is equal to the width of the DAG that gives the amount of parallelism exhibited by the job. The amount of parallelism exhibited by a resource is computed by considering the number of operations per cycle per processor, number of processors per node and number of nodes in a system. The amount of parallelism exhibited by each free resource available in the cloud is computed. The amounts of parallelism exhibited by all the available free resources in the cloud are fixed by analyzing the max, min and mid ranges. The value for the level of parallelism is assigned by comparing the amount of parallelism exhibited by each job with the max, min and mid ranges.

C. Priority Assignment

In the proposed MJHP algorithm, higher priority is given to a job which has high computational complexity and level of parallelism because time can be saved if the medium level jobs[2] are executed first. Medium priority is assigned generally to a job which needs high computational power and which exhibits high parallelism. A job, which exhibits low parallelism and needs low computational power for execution is given a low priority. The fastest free resource available in the cloud is allocated to the job which has higher priority. The procedure is given below :

Amount of Parallelism = OC* PN*NS
Where OC= No. of operations per cycle per processor
PN= No.of processors per node
NS=No. of nodes in a cloud.

Let m represent number of free resources available in the cloud and n represent the number of jobs present in the queue. The worst case time complexity of the algorithm is O(n logn) ,when m <= n and O(m logm) when m > n.

D. Proposed MJHP Algorithm

```
Step 1 : AssignLevelofParallelism( ResourceList Rs_List)
        While(Rs_List!=NULL)
        For each resource
        /*OC = No. of operations per cycle per processor
        PN = No. of processors per node
        NS = No. of nodes in a coud*/
        /* LL_List contains the amount of parallelism
        Exhibited by each resource */
        LL_List[i] = OC*PN*NS
        End While
         Find the Max, Min and Mid values in PR_List
        /* LJ_List contains the amount of parallelism
        exhibited
        by each job */
        For each job in LJ_List
        If LJ_List[i] >= Maximum
        LP_List[i] = High //LP_List contains the level of
        parallelism value
        Else If LJ_List[i] >= Middle
        LP_List[i] = Medium
        Else LP_List[i] = Low
        EndIf
        End AssignLevelofParallelism
Step 2 :Assign Priority Procedure
        AssignPriority ( CloudList CL_List)
        While( CL_List !=NULL)
        For each job
        /* CompC_List contains the Computational
        Complexity of jobs */
        If (CompC_List[i] =Medium AND LP_List[i] =
        Medium)
        Priority[i] = 1
        Else If (CompC_List[i] = Medium AND LP_List[i]
        = High)
        Priority[i] = 2
        Else If (CompC_List[i] = Medium AND LP_List[i]
        =Low)
        Priority[i] = 3
        Else If (CompC_List[i] = High AND LP_List[i] =
        Medium)
        Priority[i] = 4
        Else If (CompC_List[i] = High AND LP_List[i] =
        High)
        Priority[i] = 5
        Else If (CompC_List[i] = High AND LP_List[i] =
        Low)
        Priority[i] = 6
```

```
Else If (CompC_List[i] = Low AND LP_List[i] = Medium)
Priority[i] = 7
Else If (CompC_List[i] = Low AND LP_List[i] = High)
Priority[i] = 8
ElseIf (CompC_List[i] = Low AND LP_List[i] = Low)
Priority[i] = 9
EndIf
End AssignPriority
```

IV. PERFORMANCE STUDY

In this section, a performance study is carried out between the algorithms mentioned so far - First Come First Served scheduling algorithm (FCFS),Shortest Job Fastest Resource (SJFR), Longest Job Fastest Resource (LJFR), Min–Min algorithm, Max–Min algorithm, Priority Based Scheduling [3]and MJHP Job Scheduling algorithm.

A. First Come First Served (FCFS)

FCFS is a very basic job scheduling [7]algorithm which allocates resources to jobs as they arrive. It does not consider factors like computational complexity or level of parallelism. Hence its performance is very low. This is shown in Fig 4.

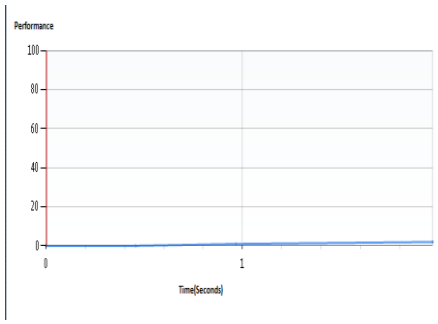

Fig 4:Implementation of FCFS

B. Shortest Job Fastest Resource (SJFR)

Shortest Job Fastest Resource is a scheduling algorithm, assigns the job with very low turnaround time to the fastest resources in the cloud. From the Fig 5 we can decipher that SJFR is more stable in handling jobs and hence outperforms FCFS scheduling algorithm.

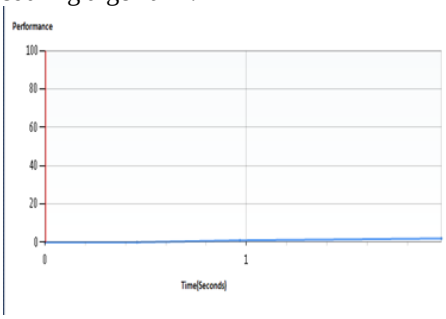

Fig 5. Implementation of SJFR

C. Longest Job Fastest Resource (LJFR)

Longest Job Fastest Resource is a scheduling algorithm that assigns the complex job to a big efficiency resource. It tries to reduce the overall execution time of the jobs. From the Fig 6 of the LJFR algorithm we can infer that LJFR outperforms FCFS and the SJFR as the jobs of high computational complexity are assigned to faster resources in the cloud which leads to shorter execution time.

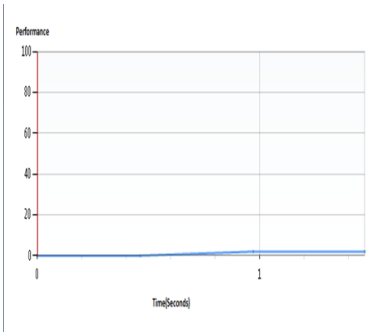

Fig 6 - Implementation of LJFR

D. Min-Min Algorithm

The Min-Min algorithm schedules the less complex jobs to high performance resources for execution. From the fig 7 we can observe that outperforms FCFS and RR but shows low performance comparing the other algorithms due to the delay caused in execution of complex jobs.

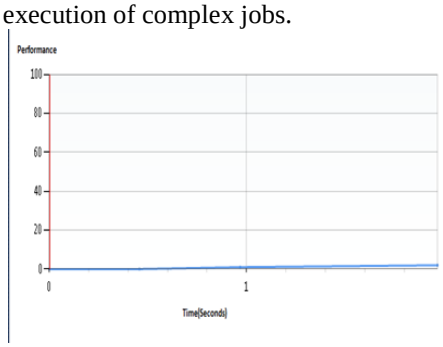

Fig 7- Implementation of Min-Min

E. Max- Min Algorithm

The complex job is scheduled first to high performance resources in the cloud and leads to the long delay in the execution of less complex jobs. Fig 8 shows the performance of the Max-Min algorithm where it outperforms FCFS, RR, MIN-MIN algorithms.

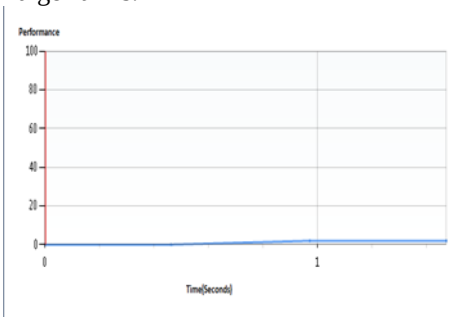

Fig 8- Implementation of Max- Min

F. Priority Based Scheduling

The priority based scheduling [3]gives high priority to a job with maximum computational complexity and level of parallelism. Even though it outperforms FCFS, SJFR, LJFR, Min-Min and Max-Min it still has a relatively low performance than MJHP job scheduling algorithm[7] as some amount of time is wasted when complex jobs are executed first.

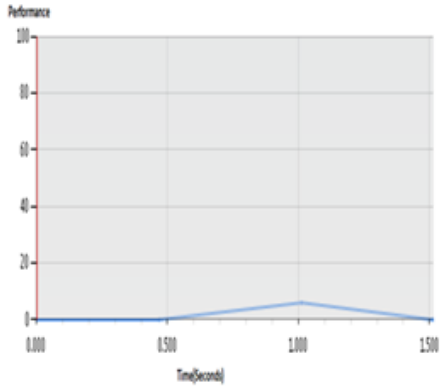

Fig 9- Implementation of Priority Based Scheduling

G.MJHP Job Scheduling

The MJHP job scheduling algorithm gives high priority to a job with medium computational complexity and medium level of parallelism. It outperforms the existing FCFS, SJFR, LJFR, Min-Min and Max-Min and the Priority Based Scheduling algorithms[3]. The MJHP job scheduling algorithm assigns the medium priority jobs with medium computational complexity and medium level of parallelism to the fastest resource . Hence the waiting time of jobs with smaller computational complexity is avoided.

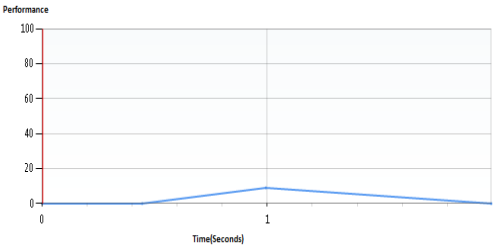

Fig 10- Implementation of MJHP Job Scheduling

H. Comparative Study Between MJHP Job Scheduling Algorithm And Other Algorithms

From the above graphs, it is evident that MJHP has a higher efficiency than other algorithms. It leads to much better utilization of [9]. Factors that govern the performance of the system such as computational complexity and level of parallelism of a resource are properly analyzed and jobs are classified as high, medium and low. First preference is given to medium level jobs as the time taken to be completed is less than high level jobs. Therefore, overall performance of the system is improved. It outperforms existing algorithms like FCFS, SJFR, LJFR, Min- Min, Max- Min and Priority Based job scheduling[3].

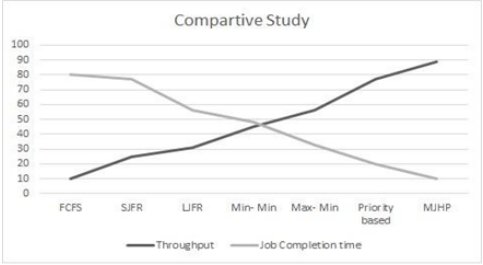

Fig 11- Comparative Study graph

From fig 11, it is notable that for an ideal cloud environment[3], MJHP scheduling proves to be the best as it provides maximum achievable throughput in a minimum amount of time making the network to work faster and more reliable.

V. Conclusion

Cloud computing[3] has evolved as a dynamic technology where more and more organizations are relying on it for various services such as email, file sharing, customer relationship management, storage and so on. Resources [9] such as servers, storage, network, applications and processes can be dynamically shaped or craved out from the underlying hardware infrastructure and made available to a workload. A proper scheduling algorithm is required for accomplishing the goals of the services. The proposed MJHP job scheduling algorithm satisfies the necessary constraints and proves to be very efficient than other existing algorithms. The reliability and consistency of the proposed MJHP algorithm is proved through simulation results and its superiority over other known algorithms is depicted.

REFERENCES

[1] S. Rekha and Santhosh Kumar. R, "Priority Based Job Scheduling for Heterogenous Cloud Environment", IJCSI International Journal Of Computer Science, Volume 11, Issue 3 May 2014.

[2] Dr. G. Sumathi, R. Santhosh Kumar, and S. Sathyanarayanan, "MidSFN Local Scheduling Algorithm for Heterogeneous Grid Environment", IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 3, No 3, May 2012

[3] Shamsollah Ghanbari, Mohamed Othman, "A Priority based Job Scheduling Algorithm in Cloud Computing", Procedia Engineering 50 (2012) 778 – 785.

[4] Isam Azawi Mohialdeen, "COMPARATIVE STUDY OF SCHEDULING ALGORITHMS IN CLOUD COMPUTING ENVIRONMENT", Journal of Computer Science, 9 (2): 252-263, 2013, ISSN 1549-3636

[5] Stelios Sotiriadis, Nik Bessis, Nick Antonopoulos, "Towards inter-cloud schedulers:A survey of meta-scheduling approaches" ,2011 International Conference on P2P, Parallel, Grid, Cloud and Internet Computing

[6] Mladen A. Vouk, "Cloud Computing – Issues, Research and Implementations",Journal of Computing and Information Technology - CIT 16,2008,4,235246doi:10.2498/cit.1001391

[7] Yun-Han Lee et al, Improving Job Scheduling Algorithms in a Grid Environment, Future Generation Computer Systems, 27(2011) 991–998

[8] Wei Wang, Cloud-DLS: Dynamic Trusted Scheduling for Cloud Computing, Expert Systems with Applications 39 (2012) 2321–2329.

[9] Tai-Lung Chen et al, Scheduling of Job Combination and Dispatching Strategy for Grid and Cloud System, GPC,(2010) 612–621.

[10] Monir Abdullah, Mohamed Othman et al, Optimal Workload Allocation Model for Scheduling Divisible Data Grid Applications,Future Generation Computer Systems 26 (2010) 971-978.