International Journal of Latest Trends in Engineering and Technology
Vol.(8)Issue(1), pp.674-679
DOI: http://dx.doi.org/10.21172/1.81.87
e-ISSN:2278-621X

A MULTI QUEUE PRIORITY SCHEDULING ALGORITHM BY USING EASY BACKFILL METHOD IN CLOUD ENVIRONMENT

Hariram Patel[1], Pankaj Richhariya[2]

Abstract- Cloud Computing is the collection of different computing resource. The aim of cloud computing is to utilize all the resource up to maximum capacity. Processing Element in cloud computing must be schedule to achieve maximum utilization. An Improved Backfill Algorithm can be used to utilize resource up to maximum capacity by using multi-level queue. Multi-level queue is used to store same priority process, in the same queue in FCFS manner and made N queue for each type of N number of priorities. The jobs are arranged according to their priorities, are merged in a single queue. The high priority queue placed first in the merged queue. After highest priority queue the second highest priority queue is placed in merged queue and third highest priority queue and so on. The resultant queue is forwarded to EASY backfill component for further processing in a sequence of priorities. The EASY schedule the given jobs in a way that high priority jobs get executed first. The combination of Improved Backfill Algorithm and EASY algorithm provides priority based efficient utilization of resources and in relatively less time than other algorithms.

Keywords- cloud computing, job scheduling, priority queue, EASY, backfill.

1. INTRODUCTION

Cloud computing is most demanding technology of this era. It provides service on demand i.e. users of cloud computing only needs to pay for what they used. This is on demand technology or platform. Customers of
Cloud Computing need not to worry about the processing instead they use services whenever they need them .A single cloud is a collection of many computing resources. And a cloud receives number of request for processing at a time. The cloud data centre receives the request from users and forward to the meta-scheduler [3] of cloud. Users' job received at Meta-scheduler are allocated to different clusters of cloud based on the information i.e. priority, Processing Element (PE) etc. The execution time of jobs is unpredictable so the scheduler schedule high priority jobs first to provide user satisfaction.
In this paper the problem of priority scheduling is addressed with higher resource utilization than other similar algorithms in cloud computing platform. There are two major part of the cloud computing one is user and other is Cloud Platform. The user wants some jobs with high priority to be executed before the other jobs and also want quality of service. The cloud platform needs to efficient so that all the resources work parallel and perform to its maximum capacity. To fulfill these entire requirements the paper proposed a way to utilize all the resources up to its maximum capacity and to execute higher priority jobs first. The combination of improved backfill algorithm and EASY algorithm [2] is increases performance of the system. Here this paper uses the similar concept with multilevel priority queue to execute higher priority jobs first. The related work of similar methods discussed in section 2. The section 3 describes problems. In section 4 implement and describe proposed algorithm and result discussed in section 5.

[1] Department of Computer Science & Engineering, Bhopal Institute of Technology & Science, Bhopal, MP, India.
[2] Department of Computer Science & Engineering, Bhopal Institute of Technology & Science, Bhopal, MP, India.

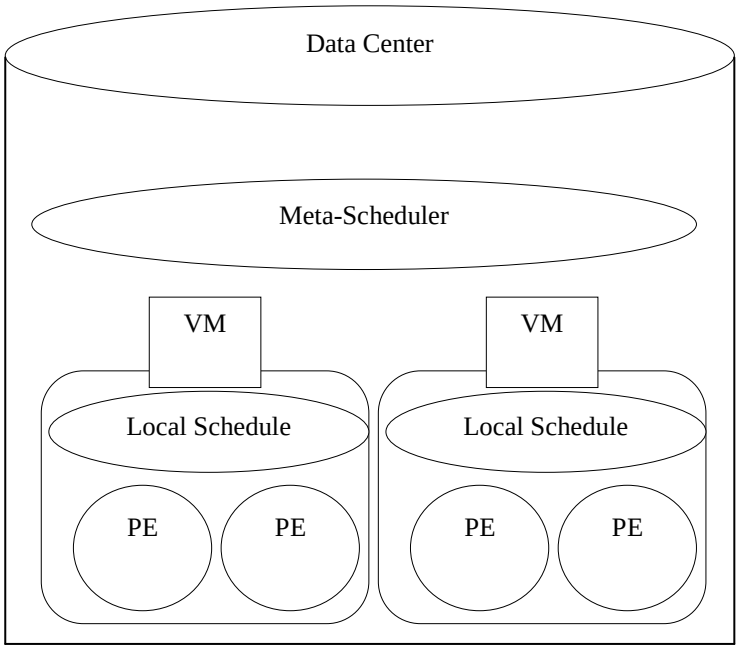

Figure. 1. Cloud Data Center and Meta-scheduler

2. LITERATURE REVIEW

The job aim of job scheduling is to improve resource utilization with quality of services. In cloud computing the scheduling of Meta-scheduler should satisfy both cloud service provider and users' of cloud. There are many algorithms for scheduling jobs and many of them also applicable in cloud platform.

The simplest algorithm of job scheduling is First-Come-First-Serve(FCFS)[8].In this algorithm jobs picked from the job queue by Scheduler, the job arrived first placed on front of queue and followed by other jobs according to their arrival in the system. This scheduling leads to starvation when a large job comes before small jobs. The FCFS doesn't deal with priority of jobs.

The starvation problem dealt by Shortest Job First(SJF)[9] method. In this method the waiting queue is sorted according to their processing needs. The smallest job is placed on front of queue. This algorithm reduces the waiting time all the jobs. This algorithm also suffer starvation problem , if there are large number of small jobs the large job must wait and it may be infinite if jobs small jobs continue coming in the system. The another is Longest job first[9] algorithm. This algorithm also suffers from problem of starvation.

These algorithms fail to efficiently utilize cloud resource and not provided any satisfaction in execution time also. There is an another efficient algorithm called Backfill scheduling algorithm[5] based on FCFS[8] provide better utilization of resources.

Another algorithm to use resources efficiently is the Extensible Argonne Scheduling sYstem (EASY)[10] algorithm. This algorithm is based on backfilling concepts. This algorithm is executed FCFS basis and picks all the resources those are not assigned to any job. It assigns the jobs to the idle resources if fit on it. By using this backfill method resource utilization is increase. This algorithm does not provide any priority based concept.

For improving quality of services and resource utilization improved backfill algorithm (IBA)[2] has been proposed. This algorithm uses Balanced Spiral (BS) method to schedule jobs in "V" shape sequence. It improves utilization of resources to a large extent. The IBA proves efficient in static environment but in dynamic process comes with priority. Hence the scheduling algorithm must deal with the priority of jobs[6].

3. PROPOSED MODEL

In cloud computing, users' requirement and resource utilization both are very important. Scheduling algorithm based on size of jobs fails in real scenario, because in real cloud environment no one can predict the execution time of jobs. On The other hand ,to satisfy user with services, cloud platform must consider the priority of jobs. The high priority jobs must be executed first. EASY algorithm are executed on first come first serve basis. The proposed method also takes advantage of priority scheduling in IBA and EASY algorithms.

3.1 Proposed model

The user needs to submit their job with given priority at data centre. The Data centre estimates the required Processing Elements (PE) for the given jobs. Here we consider that the execution time is same for all jobs (Unit 1 is taken) and all the

jobs are independent of each other. The Data Center also assign priority (1 to 3 in our case, 1 is highest and 3 is lowest) from the given range. There are n queue for n priority jobs. All the jobs placed in waiting queue according to the arrival time.
Once all jobs are placed in waiting queue[8], they are sorted according to priority assigned. Same priority jobs kept in same queue and all the jobs sorted in FCFS basis inside the priority queue. Now merge all the priority queue to form a single long queue which comprises all the queue arranged according to their priority.

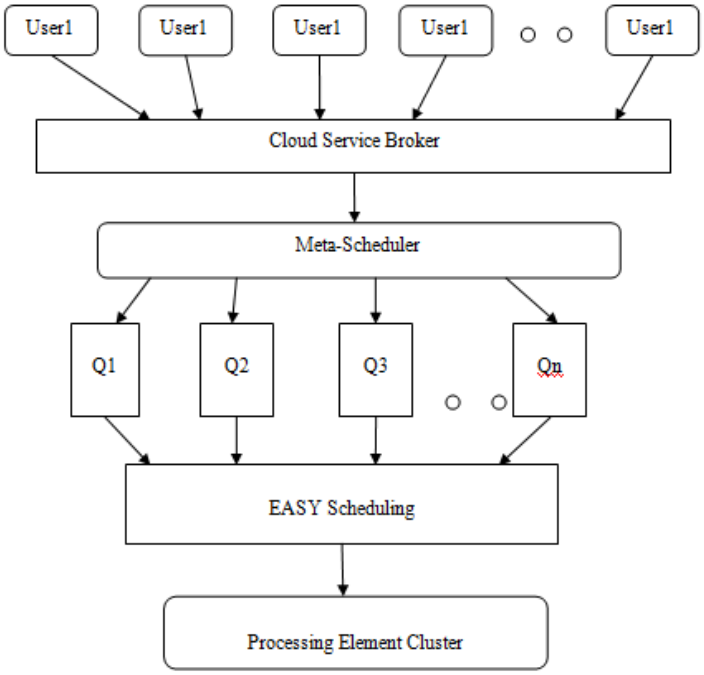

Figure.2. Proposed method to of Priority Scheduling

By arranging queue according to priority, the high priority queue placed first followed by second highest and so on. The lowest priority jobs placed last in the queue. The sorted queue is now taken as a input to EASY algorithm. An EASY algorithm uses the resources efficiently to maximize throughput of the system and minimize the execution time.

3.2 Details of Algorithms

Our algorithm has three major steps:-

1. All the jobs sorted and stored in different multi-level priority queue as same priority jobs placed in same queue.
2. Combine the resultant job sequence into a single queue.
3. Now apply EASY backfill algorithms to obtain final result.

Input:

- Queued jobs with nodes
- Number of free nodes

Algorithm EASY backfill:

1. Find extra nodes

(a) Find when enough nodes will be available for the first queued job.

(b) If this job does not need all the available nodes, the ones left over are the extra nodes.

2. Find a backfill job

(a) Loop on the list of queued jobs in order of arrival.

(b) For each, check whether it requires no more than the extra nodes.

(c) The first such job can be used for backfilling.

Figure. 4. EASY Algorithm

The result of Step 3 is final. The meta-scheduler selects jobs from final result and executes them in the given order.

3.3 Example to show results of proposed method

Let there is 15 jobs in submitted to cloud data center. All the jobs having priority associated with them and number of processing elements required to execute the given jobs. Assume that processing time is equal for each element and all jobs are arrived at same time.

Jobs	**PE Required**	**Priority**
J1	5	1
J2	6	2
J3	2	3
J4	4	2
J5	9	3
J6	1	1
J7	7	3
J8	3	1
J9	8	2
J10	5	1
J11	6	2
J12	2	3
J13	4	2
J14	9	3
J15	1	1

Table 1. Jobs with required Processing Elements and Priority

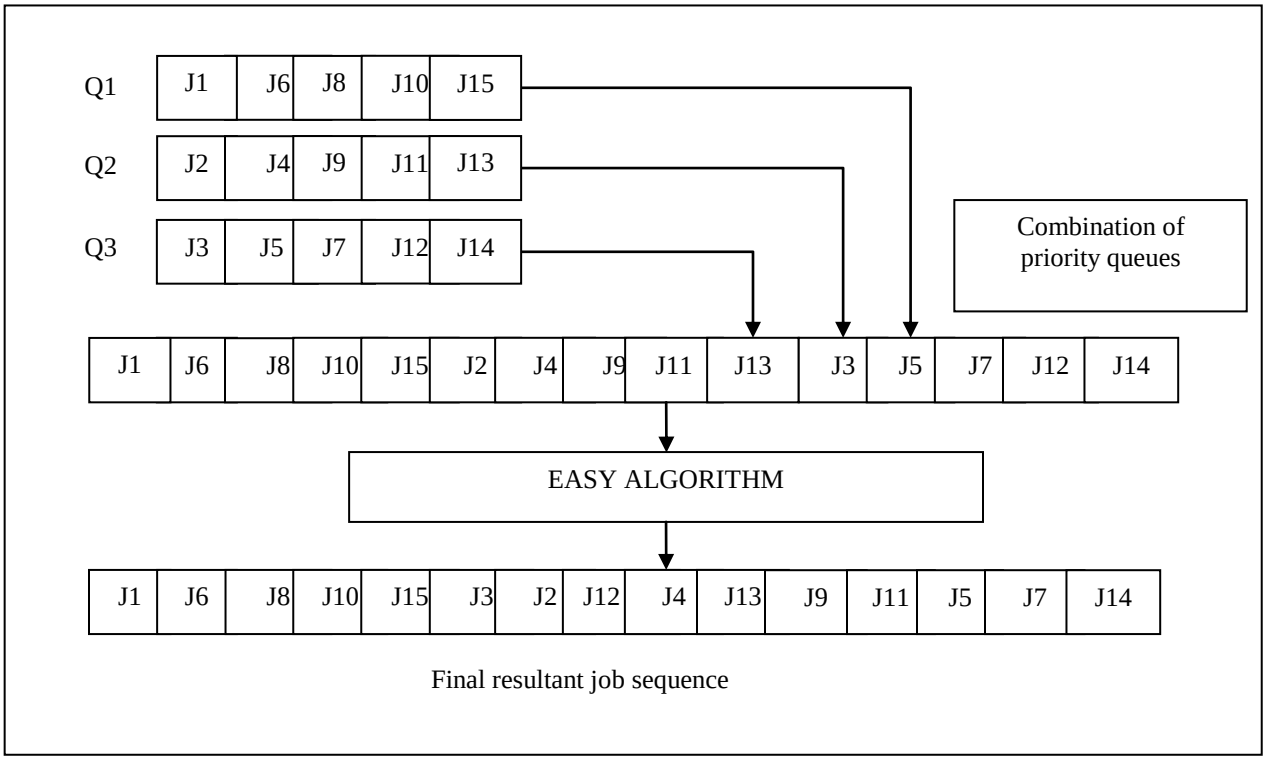

Figure 5: Over-all working of MQPS

<table>
<tr><td rowspan="10">PROCESSORS</td><td>PE9</td><td rowspan="3">J8</td><td></td><td></td><td></td><td></td><td rowspan="2"></td><td rowspan="9">J5</td><td rowspan="2"></td><td rowspan="9">J14</td></tr>
<tr><td>PE8</td><td rowspan="2">J3</td><td rowspan="2">J12</td><td rowspan="3">J13</td><td rowspan="8">J9</td></tr>
<tr><td>PE7</td><td rowspan="8">J7</td></tr>
<tr><td>PE6</td><td>J6</td><td>J15</td><td rowspan="6">J2</td><td rowspan="2">J11</td></tr>
<tr><td>PE5</td><td rowspan="5">J1</td><td rowspan="5">J10</td></tr>
<tr><td>PE4</td><td rowspan="3">J4</td></tr>
<tr><td>PE3</td></tr>
<tr><td>PE2</td></tr>
<tr><td>PE1</td></tr>
<tr><td colspan="10">TIME</td></tr>
</table>

Figure.6. Processor allocation table

4. RESULT AND DISCUSSION

In this section, the experimental evaluation of the proposed method is discussed. CloudSim toolkit is used to simulate the proposed method with different experimental set-ups. The classes available in CloudSim are extended to implement the proposed method and other job scheduling strategies.

Basically, performance of Multi-Level Priority Scheduling (MLPS) is evaluated on the basis of two measures.

- Resource utilization and
- Processing time.

The result of proposed algorithm is compared with the other similar algorithms like FCFS,EASY and IBA algorithms.

Resource Utilization =Resource Used/Total Available resources.

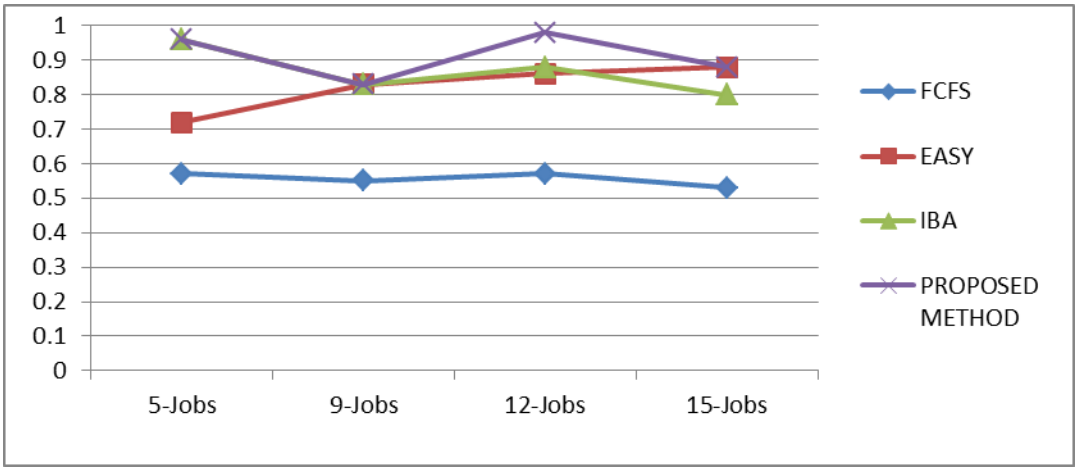

Figure.7.Resource Utilization Comparisons

In experimental evaluation we have analyzed 5 different examples of varying range of number of jobs (5, 9, 12, 15, 20). The algorithm proves better from all the similar jobs in all cases in terms of resource utilization and execution time as shown in Figure. and Figure. . The algorithms is better than similar algorithms and also handle priority of jobs. Handling priority of jobs make it user satisfactory. Hence the ultimate goal of resource utilization with minimum time and user satisfaction is achieved by the algorithm.

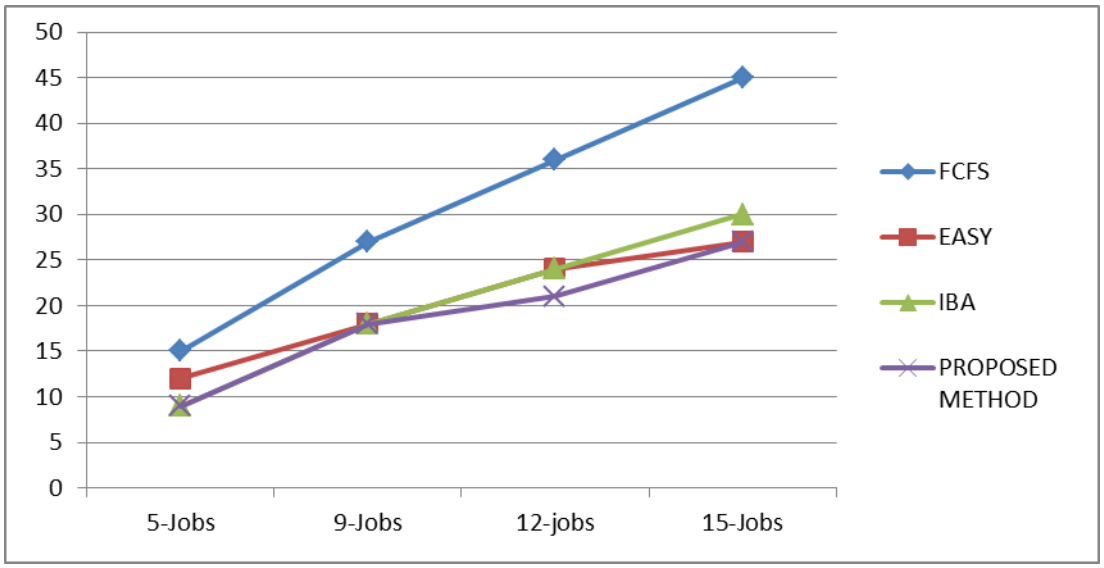

Figure.8. Execution Time Comparison

5. CONCLUSION AND FUTURE WORK

Multi-level priority queue scheduling algorithms which uses IBA and EASY backfill algorithm to achieve maximum utilization of resources have proves better from other similar algorithms. This algorithm also provide chance to execute high priority jobs before the lower priority jobs. The ultimate goal of priority scheduling is achieved and no loss in resource utilization rate and execution time. Hence the integration of all these strategies increases the overall efficiency of the system with user satisfaction.

The proposed algorithm provide a better result in ClouSim simulator but in real cloud environment the results may be different. In real cloud platform the process comes on system at random time with different priorities. Sorting of job in dynamic environment is difficult. Here we also consider execution time is same for all job but in real it varies from job to job. Our proposed model is need to be upgraded to handle jobs of different execution time and to handle all jobs coming at random time in system.

6. REFERENCES

[1] Dubey Kalka,Kumar Mohit,Chandra Mayank Arya "A priority based job scheduling algorithm using IBA and EASY algorithm for Cloud Meta-scheduler", International Conference on Advances in Computer Engineering and application(ICACEA) IMS Engineering College Ghaziabad,India 2015.

[2] Suresh.A, Vijayakarthick.P ,"Improving scheduling of backfill algorithms using balanced spiral method for cloud meta-scheduler", IEEE-International Conference on Recent Trends in Information Technology, ICRTIT, June 2011.

[3] M. Peixoto, M. Santana, J. Estrella T. Tavares, B. Kuehne, and R.Santana, "A meta-scheduler architecture to provide QoS on the cloud computing", in Telecommunications (ICT), IEEE International Conference on, pp. 650 657, April 2010.

[4] R. Buyya, R. Ranjan, and R. Calheiros, "Modeling and simulation of scalable cloud computing environments and the cloudsim toolkit:challenges and opportunities", in High Performance Computing Simulation, International Conference on, June 2009.

[5] S. Yi, Z. Wang, S. Ma, Z. Che, F. Liang, and Y. Huang "Combinational backfilling for parallel job scheduling," in Education Technology and computer (IECTC),june 2010.

[6] S. Ghanbari and M. Othman, "A priority based job scheduling algorithm in cloud computing ," Procedia Engineering,International Conference on Advances science and Contemporary Engineering 2012.

[7] A. Tripathi and A. Mishra, "Cloud computing security considerations," in signal Processing, Communications and Computing, IEEE, International Conference on,2011.

[8] X. Xu and N. Ye, "Minimization of job waiting time variance on Identical parallel machine", Systems, Man, and Cybernetics, Part C: Applications and reviews, , IEEE Transactions on, Sept. 2007.

[9] J. Yang and Z. Chen, "Cloud computing research and security issues", in Computational Intelligence and Software Engineering(CiSE),International conference on, 2010.

[10] Adam K.L. Wong and Andrzej M. Goscinski, "Evaluating the EASY-Backfill Job Scheduling of Static Workloads on Clusters", IEEE International Conference on Cluster Computing,2007.

[11] J. Ru and J. Keung, "An Empirical Investigation on the Simulation of Priority and Shortest-Job-First Scheduling for Cloud-Based Software Systems," Software Engineering Conference (ASWEC), Melbourne, 2013.