International Journal of Applied Engineering Research ISSN 0973-4562 Volume 12, Number 9 (2017) pp. 1929-1934

An Smarter Multi Queue Job Scheduling Policy for Cloud Computing

Jaspreet Singh[#1] and Deepali Gupta[#2]

[1] *Ph.D Research Scholar, Department of Computer Science and Engineering, Maharishi Markandeshwar University, Sadopur, Ambala, Haryana, India.*

[#1]ORCID: 0000-0002-5499-3189

[2] *Professor & Head, Department of Computer Science and Engineering, Maharishi Markandeshwar University, Sadopur, Ambala, Haryana, India.*

[#2]ORCID: 0000-0002-3207-5248

Abstract

Cloud computing is an extension paradigm of grid and distributed computing. Cloud providers mainly focus on managing computing power, energy consumption, storage and services that are assigned to external users via internet. Managing users requirements have created several challenges in optimize job scheduling and on-demand resource allocation. Cloud job scheduling can be viewed as NP-hard optimization problem. An efficient job scheduler should choose scheduling policy to increase the performance of system. In earlier research an efficient multi queue scheduling (MQS) algorithm was build which divide user jobs in multiple queues and carry out dynamic selection of user jobs for execution. It successfully plummet the problem of fragmentation associated with the tradition job scheduling algorithms like First Come First Serve, Round Robin etc but left behind some drawbacks of higher switching time between multiple queues and dynamic selection posses high probability of indefinitely postponement of different types of user jobs causing long job waiting time therefore results in higher energy consumption. To address this issue, inspired by the concept of multi queue scheduling we introduce a Smarter MQS model which effectively separate user jobs into two job queues then give more importance in formation of merging jobs pattern by merging user tasks from both queues for execution, so the technique will empower us to reduce energy consumption while naturally to some degree will reduce job completion time and the overall cost. The proposed technique will achieve a high degree of job scheduling in cloud computing environment.

Keywords: Scheduling in Cloud Computing, Energy Efficient Multi Queue Scheduling, Smarter Multi Queue Job Scheduling in Cloud Computing**.**

INTRODUCTION

Cloud computing offers coherent scalability and elasticity of IT resources that are provided to the end users via internet. It started and emerged as a popular risk free concept amongst end users as the somebody else is setting up all the infrastructure while end users on small and large scale can easily use the complex infrastructure while paying for only what they use [1].Cloud computing environment provides several distinct workflow features, all of which cannot be found in one computing environment. For example, standardized access over networks, determination of type and number of resources through service requests, possibility of elastic scalability of the resources provided to end user and on-demand services [2]. Clouds are much more dependable platform than grids, they are cheap and scalable alternative as compared to specialized clusters and supercomputers [3]. Clouds offers virtual services (hardware, user-interface, database, application logic) so that users present anywhere in the world get a chance to deploy applications on-demand at a very competitive cost based upon the quality of service (QOS) demanded by them [4]. As in cloud environment various cloud services are shared among the millions of end users so scheduling in cloud computing can be expressed as cycle of decisions regarding the allocation of millions resources to millions of computing jobs submitted by the end users. As mentioned by authors of [5] a good scheduling strategy would perfectly serve for hassle-free load balancing and allow optimized utilization of basic resources. The current scheduling algorithms work great theoretically as well as during their use on single machine but with rise in cloud capabilities, use and requirements, they will fail drastically. In real world, there is no single machine. There are multiple job processing done on multiple machines and all of them work in parallel rather than as single entities. Virtual machines [6] assigned to user jobs based upon the characteristics of the jobs. The single job constitute series of task [7].The scheduler should consider several parameters like queue type and length, resource capabilities (clock speed, memory size) etc to increase the throughput of the system [8]. A scheduler often aims at achieving several goals such as minimization of response time, reduction in energy consumption, enhancing throughput and maximizing fairness and we need all of them to be on the first bench and be dealt with equal priority. But in practice, these goals almost always create a conflict leading to a compromised rather than optimized result. Only one of the above mentioned four goals are given preference depending upon user purpose and need.

KEY CONCEPTS AND TERMINOLOGIES

Architecture of Cloud Computing :

The four-layer architecture for cloud computing can be explained as [9]:

Fabric layer: It holds all the resources related to hardware level such as compute resources (deals with processing capability of machines), network resources (deals with connections within network connecting machines) and storage resources (deals with storage capability of machines) [9].

Unified Resource Layer: The detail of large number of resources working in collaboration with one another to make a powerful network are maintained by Unified Resource layer , implementation of resources is abstracted from the end users by virtue of virtualization [9].

Platform Layer: Platform layer constitute of services, specialized tools and middle ware to provide a platform of deployment and development like scheduling service, web hosting environment etc [9].

Application Layer: It accommodates the applications that will run in the Clouds [9].

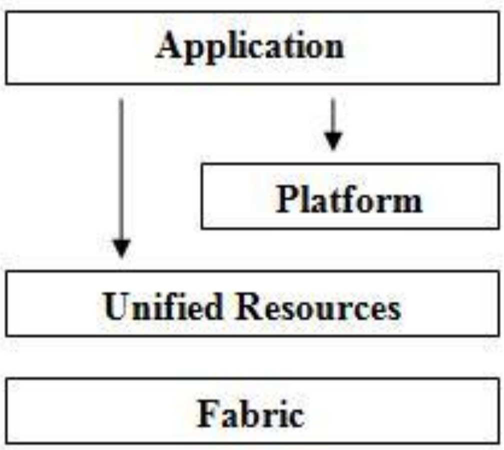

Figure 1: Cloud Architecture

Related Work :

Job scheduling can be defined as arduous task because scheduler performs an challenging work, it have to order the user jobs while side by side have to maintain an equality between QOS, efficiency and fairness [10]. As author Pinal Slot [11] done the study about various existing scheduling algorithms which yield that as in First come first serve (FCFS), the jobs are processed according to their time of arrival. There is no prioritization and it often leads to missing deadlines and long waiting periods for otherwise urgent and top-priority tasks. The Round Robin (RR) scheduling is great at improving the average response time. However, as waiting time is dependent on number of processes, it fails when the number increases. In priority scheduling waiting time and response time is different for each process depending upon its priority level. Higher priority processes have significantly smaller waiting times, but long waiting queues are bound to exist and affect the performance while the Shortest Job First Serve scheduling is the special case of priority scheduling in which longer processes get affected because turnaround time is calculated by taking into account waiting time and processing time. In [12] AV.Karthick, proposed a multi queuing model which reduces the problem acknowledged by existing scheduling method. It reduces the starvation to much extent with in processes by doing the clustering of user jobs based upon burst time.

In Wanqing You [5] work, based upon the concept of multi queue they proposed new scheduling strategy that considers several factor like different configuration of virtual machines, history log and priority of various user task. The author Iqra Sattar [13] in his work proposed an MLQPTS (Multilevel Queue with Priority & Time Sharing Scheduling) algorithm in which depending upon characteristics of process the priority level the user task is set and they are assigned in the queue. The queue is executed for a specific time and a new queue is formed for next round the method which improves starvation but come out with the concern related to CPU utilization.

Problem Description :

The authors of [12] have proposed an efficient multi queue job scheduling for the cloud computing in which client job are categorize into small job , medium job and large job then dynamic selection of jobs are done to reduce the processing time and waiting time for the process in the queue. But as we investigated in this algorithm the client jobs are categorized in three queues so it will directly affects the switching time needed by the scheduler to switch from one job to another, secondly in this algorithm dynamic selection of jobs from different queues is carried out which have the high probability that in each cycle scheduler can select the user process from short job queue tend to delay the several processes in medium or large queue and vice versa. Both factors tend to affects the energy consumption and job completion time in cloud environment.

Proposed Smarter MQS Approach for the Cloud Computing Environment :

(A) Smarter MQS Architecture

This paper proposes a Smarter Multi queue Job Scheduling algorithm to schedule user jobs and to recognize the status of all virtual machines in real time environment in cloud computing.

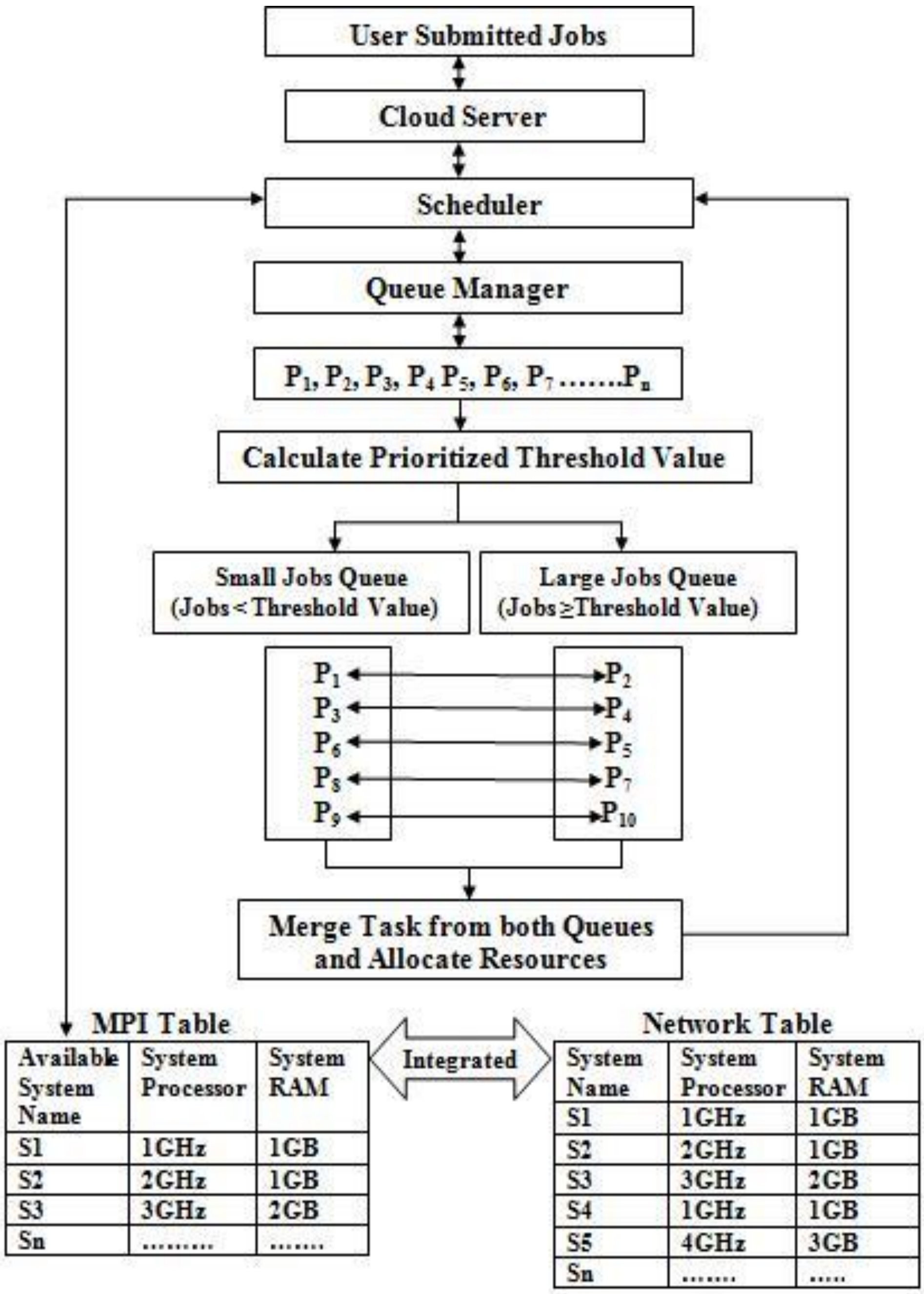

Figure 2: Architecture of Smarter MQS for Cloud Computing.

Based on received user jobs, the algorithm actively separate the unscheduled jobs to multiple queues then formulate merge pattern by merging the one job from the small job queue with the one job contained in large job queue to avoid a long idle time by jobs in the queues. Further scheduler dispatches the merge task set for execution.

The core components which are used in above architecture are discussed below:

1. **Network Table:** The cloud environment dispense heterogeneous resources with varying processor speed so a network table constitutes a set of heterogeneous resources. Resources are usually pooled to cater to several consumers with dynamic allocation and assignment on the basis of consumers demand. A multi-tenant model will be followed in the process.

2. **MPI Table**: The scheduler uses an MPI table that's synchronized to it so that the jobs forwarded by scheduler are scheduled via this table which further works in collaboration with the network. The Message passing interface table acknowledges all the vacant system from network and maintains the array of all these available systems for processing.

3. **Scheduler:** The main purpose of implementation of scheduler is to utilize all the network resources efficiently and cater to multiple individuals in an effective and optimized manner. Using an MPI table in the suggested smarter MQS technique, it will ensure that all the merge task set allocated to virtual system and all resources should be utilized in an optimal way.

(B) How it works?

The Figure 2 depicts the Smarter MQS algorithm for scheduling user jobs. The algorithm consists of the following steps:

Step 1: The Scheduler in collaboration queue manager accepts the unscheduled job from multiple users for processing. It further provides resources for the processing of these jobs. The Queue Manager directs the scheduler about the output of the processed jobs. The Scheduler is directly linked to MPI table.

Step 2: According to the projected energy required by each process a prioritized threshold value is calculated. Now keeping this value as base value the jobs are categorized into small and large queue jobs.

Step 3: As the jobs has been categorized in different queues so as to reduce the idle job waiting time, merging pattern will be formed by merging of user processes from both the queues. The pattern can have multiple set of merge task. By doing this each process is allocated to resource in the desirable amount of time and very less probability that deadlines for several process completions will be missed.

Step 4: A single set of merge user task from merge pattern is picked up and will be allocated by scheduler to the available system looking at the list maintained by MPI Table.

(C) Strengths of Smarter MQS

The major benefits provided by this Smarter MQS Architecture are as follow:

1. **Lesser Switching Time:** It is the time taken for switching from one process to another. Unlike popular belief, it isn't a one-step task as it includes several tasks that have to be done by administration like saving and loading of registers, memory maps and so on. The switching time is higher in usual algorithms due to long waiting queues but in our proposed smarter MQS algorithm, as it is dealing with two queues instead of three queues (as used in MQS architecture build in [12]) so the switching time, as a result, will be significantly reduced.

2. **Reduce Job Completion Time**: It is the amount of time a job takes to execute completely. As the scheduler of our discussed technique combines the jobs from both the queues we are using, resulting in all the jobs being executed without waiting for turns, thereby it will be reducing job completion time to some extent. Also for the jobs the time that is spent waiting in queues contributes to job idle waiting time, the proposed approach discussed in this paper will fixes this problem too. Traditional scheduling methods also cause low throughput due to high response times.

3. **Easy Jobs Allocation**: There is large number of user who submits their jobs in cloud environment so the scheduler aims to control degree of multiprogramming. It selects ready jobs from the queue and submits them to various virtual systems for processing. As in our proposed technique, scheduler is directly synchronized with the MPI table which maintain the list of available virtual systems as a result no PING command will be issued by the scheduler side to the Network(contains set of virtual systems) because it selects the system from the list maintained by MPI table. By this method, the workload on the behalf of scheduler gets reduced and so the energy.

4. **Minimize Starvation:** As the scheduling policy framed in our proposed smarter multi level queue scheduling aims to cover the indefinitely postponement of the user process so no user process will go in the freeze state. The probabilities of getting virtual systems for each type of user jobs are same

so resources will be allocated to user process in a fair manner.

Scheduling Strategy :

In the high performance computing community scheduling of parallel jobs on cluster and supercomputer proved to be an active research topic [14]. Job Scheduling algorithm in Cloud is subdivided in two main parts: Batch Mode Heuristic Scheduling Algorithm (BMHA) other is On-Line Mode Heuristic Scheduling Algorithm. The basic difference between the two is that in BMHA as the job arrived they are queued and after fix period of time the scheduling algorithm start while in other as the jobs arrive to system they get queued for processing. FCFS, Round Robin, Min-Min, Max-Min algorithm are the example of BMHA scheduling algorithm [15]. The study of scheduling on cloud is considered as one of the quality study done by the researchers although many studies explained the scheduling on cloud in simple and clear form, there still exist diverse definitions [16].

Job scheduling algorithms is one of the most challenging hypothetical problems in the cloud computing domain area [17]. The scheduling problem aim to solve in this paper can be stated as:

In order to map all the Jobs(J) of Workflow(W) to a set of Cloud services (S). $S = \{s_1, s_2, \ldots, s_x, \ldots, s_z\}$ is the set of available computing cloud services with varied processing capability. The Scheduler utilizes this S in such a way, that the estimated total energy and time incurred for processing the user jobs should be minimized.

While Makespan(W) ≤ Deadline ;

Total Energy (W) Consumed and Cost < Previous Efficient MQS Technique (used in [12]) alongside satisfying all the constraints).

CONCLUSION AND FUTURE WORK

A smarter multi queue job scheduling policy for cloud computing is proposed in this paper which divide the cloud users jobs in multiple queue and then merging the jobs from both queues carry out optimal allocation of these jobs to available virtual systems by referring to MPI table. The discussed policy will aim to reduce the like hood of delaying user jobs thereby it will sharply reduce energy consumption and to some extent limit jobs completion time. In future the work can be carried out on optimization of merging pattern of user jobs so to decrease more energy and time thereby decreasing the load on scheduler.

REFERENCES

[1] R. Buyya, “Introduction to the IEEE Transactions on Cloud Computing”, IEEE Transactions on Cloud Computing, vol. 1, no. 1, 2014, pp. 3-9.

[2] Jianfeng Zhao; Wenhua Zeng; Min Liu; Guangming Li; Min Liu, “Multi-objective optimization model of virtual resources scheduling under cloud computing and it’s solution” , Cloud and Service Computing (CSC), 2011 International Conference, 12-14 Dec. 2011, pp.185-190.

[3] A. Iosup, S. Ostemann, M. Yigitbasi, R. Prodan, T. Fahringer, D. Epema, “Performance analysis of Cloud computing services for many-tasks scientific computing” IEEE Transaction on Parallel Distributed Systems, vol 22, no6, (2011), pp. 931–945.

[4] Rajkumar Buyya, R. Ranjan and Rodrigo N. Calheiros, “Modeling and Simulation of Scalable Cloud Computing Environments and the CloudSim Toolkit: Challenges and Opportunities” in the 7th High Performance Computing and Simulation Conference (HPCS) IEEE Computer Society, 2009, DOI: 10.1109/HPCSIM.2009.5192685, pp. 1-11.

[5] W. You, K. Qian and Y. Qian, "Hierarchical Queue-Based Task Scheduling," Journal of Advances in Computer Networks, vol. 2, no. 2, June 2014.

[6] Saraswathi AT, Kalaashri.Y.RA, Dr.S.Padmavathi, “Dynamic Resource Allocation Scheme in Cloud Computing”, Procedia Computer Science 47 (2015) 30 – 36, DOI: 10.1016.

[7] Tingting Wang, Zhaobin Liu, Yi Chen, Yujie Xu, Xiaoming Dai, “Load Balancing Task Scheduling based on Genetic Algorithm in Cloud Computing”, IEEE 12th International Conference on Dependable, Autonomic and Secure Computing, 2014, pp. 146-152.

[8] R. Buyya, D. Abramson, J. Giddy. “Nimrod/G: An Architecture for a Resource Management and Scheduling System in a Global Computational Grid”, IEEE International Conference on High Performance Computing in Asia-Pacific Region (HPC ASIA) , 2000.

[9] I. Foster, Y. Zhao, I. Raicu, and S. Lu, “Cloud computing and grid computing 360-degree compared”, Grid Computing Environments Workshop, 2008. GCE’08, 2008, pp. 1-10.

[10] Rutvij H Jhaveri,Yash P. Dave,Avani S. Shelat, “Various Job Scheduling Algoritms in Cloud

Computing: A Survey" ,Information Communication and Embedded Systems (ICICES), International Conference 2014,1-5,DOI 10.1109.

[11] Pinal Salot, "A Survey of various Scheduling Algorithm in Cloud Computing Environment," International Journal of Research and Engineering Technology (IJRET), Vol. 2, 2013, pp.131–135.

[12] A. V. Karthick, E. Ramaraj, and R. G. Subramanian, "An Efficient Multi Queue Job Scheduling for Cloud Computing", In Computing and Communication Technologies (WCCCT), 2014 World Congress, February 2014, pp. 164-166.

[13] I.Sattar, H.Shahid and N. Yasir, "Multi-Level Queue with Priority and Time Sharing for Real Time Scheduling", IEEE International Journal of Multidisciplinary Sciences and Engineering, vol 5, no 8, August 2014.

[14] J. Cao, F. Zimmermann, "Queue Scheduling and Advance Reservations with COSY". Proc. 18th IEEE Int. Parallel & Distributed Processing Symp., Santa Fe, NM, USA 2004, p. 63.

[15] S. Ghanbari and M. Othman, "A priority based job scheduling algorithm in cloud computing," in Proceedings of the International Conference on Advances Science and Contemporary Engineering, October 2012, pp. 778–785.

[16] C. W. Tsai and J. Rodrigues, "Meta heuristic Scheduling for cloud: A survey", IEEE Systems Journal, vol. 8, no. 1, 2014, pp. 279–297.

[17] Abhishek Gupta, H.S Bhadauria et al, "A Theoretical Comparison of Job Scheduling Algorithms in Cloud Computing Environment", NGCT, September 2015, DOI: 10.1109, pp. 16-20.

[18] Jaspreet Singh, Deepali Gupta, "Energy Efficient Heuristic Base Job Scheduling Algorithms for Cloud Computing", IOSR Journal of Computer Engineering (IOSR-JCE), Volume 19, Issue 1, (Jan.-Feb. 2017), DOI: 10.9790/0661-1901015055, pp. 50-55.