AENSI Journals

Australian Journal of Basic and Applied Sciences

ISSN:1991-8178

Journal home page: www.ajbasweb.com

Real Time Online Job Scheduling Algorithm with Manifold Resources for Cloud Computing

[1]R. Santhosh and [2]T. Ravichandran

[1]*Research Scholar, Department of Computer Science and Engineering, Karpagam University, Coimbatore, India.*
[2]*Principal, Hindusthan Institute of Technology, Coimbatore, India.*

ARTICLE INFO

Article history:
Received 19 August 2014
Received in revised form
19 September 2014
Accepted 29 September 2014
Available online 3 November 2014

Keywords:
Nonpreemptive,
Migration,
Checkpoint, Deadline,
Average Waiting Time.

ABSTRACT

Cloud Computing is the advanced technology which is developing radically to provide the services over the internet. Job scheduling is one of the major activities performed in the cloud environment. In older approaches a non pre-emptive real time scheduling using checkpointing algorithm is used to minimize the penalty and the execution time of the migrated jobs through by allocating checkpoint intervals to the job during their execution. In this algorithm, the jobs are executed with a single resource (SR) in a non pre-emptive manner. Therefore it increases the average waiting time of a job in the queue. In order to provide solution to this problem, we proposed a real time online job scheduling algorithm with manifold resources (MR) in the cloud environment. This proposed algorithm uses manifold resources for job execution in a non pre-emptive fashion and decreases the average waiting time of a job in the queue. Our simulation results show better performance than the existing approaches based on the similar model.

© 2014 AENSI Publisher All rights reserved.

To Cite This Article: R. Santhosh and T. Ravichandran, Real Time Online Job Scheduling Algorithm with Manifold Resources for Cloud Computing. ***Aust. J. Basic & Appl. Sci.,*** *8(17): 443-446, 2014*

INTRODUCTION

Cloud computing is the internetworking based computing where many services are given by the cloud providers. They offer these services in the form of private, public and hybrid cloud. The users have to pay for the used services. In this cloud computing environment, users are actively consuming all the services in a functional and effective manner. Scheduling of a job plays a key role to acquire more profit. Cloud consumers should avail these services properly by scheduling and completing the job within the deadline.

Non pre-emptive real time scheduling using checkpointing algorithm provides the solution for minimizing the execution time of the migrated jobs. Checkpoint intervals are allocated for the jobs. Whenever migration takes place due to missed deadline, the job will be executed from the last checkpoint interval saved, Santhosh and Ravichandran (2013).

In this paper, we present a Non Pre-emptive manifold resources scheduling Algorithm. Our algorithm decreases the average waiting time of a job in the queue and improves overall efficiency of the system.

In this section, we present a Non Pre-emptive manifold resources scheduling Algorithm to decrease the average waiting time of a job in the queue. Two different queues are used namely standard queue and migration queue. Job Scheduler is responsible for allocating the resources to the job. More resources will be allocated to process the standard queue rather than migration queue.

Non Pre-emptive manifold resources scheduling Algorithm:

Step 1: Let Q= {C_1, C_2, C_3....C_n} be the various jobs to be processed in the standard queue.
Step 2: Sort the jobs in the queue based on the deadline.
Step 3: Checkpoint intervals will be allocated for the jobs present in the standard queue.
Step 4: Resource table is generated by the job scheduler for the available resources.
Step 5: The job scheduler allocates the needed resources for the jobs in the standard queue.
Step 6: The new job is ready to process by the job scheduler from the standard queue only if the standard queue and the migration queue is empty.
Step 7: All the available resources will be used to process the various jobs by the job scheduler from the standard queue only if there is a load in the standard queue and the migration queue is empty.

Corresponding Author: R. Santhosh, Research Scholar, Department of Computer Science and Engineering, Karpagam University, Coimbatore, India.

Australian Journal of Basic and Applied Sciences, 8(17) November 2014, Pages: 443-446

Step 8: All the available resources will be used to process the job by the job scheduler from the migration queue only if there is a load in the migration queue and the standard queue is empty.
Step 9: If both the standard and the migration queue are loaded, the job scheduler allocates more systems to process the standard queue and few systems to process the migration queue.
Step 10: When the job misses its deadline during execution, the job scheduler place the job in the migration queue. The migrated job will start its execution from where the last checkpoint interval saved.
Step 11: If the job placed in the migration queue does not have enough resources to process, then the job scheduler checks for the available resources in the systems allocated to the standard queue.
Step 12: If the job needed resources is matched with the systems, then the job scheduler places that job again in the standard queue and executed only when the resources gets free.
Step 13: The job processed in the standard queue gets more preference than the job processed in the migration queue.
Step 14: If the job in the standard queue does not have enough resources to process, then the job scheduler checks for the available resources in the systems allocated to the migration queue.
Step 15: If the job needed resources is matched with the systems, then the job scheduler places that job in the migration queue and executed immediately if the required resources is free otherwise the job will be executed by suspending the currently executing job. Then the suspended job will be moved and placed again in the migration queue.
Step 16: Whenever the resources get free, the moved job in the migration queue starts its execution again from the last checkpoint interval saved.

Whenever the job misses its deadline, the job will me migrated by suspending its execution. The communication channels are temporarily redirected. The job will be removed from the source node and place it in the migration state. Migration process gets completed when all the required states are transferred from the source node to the destination node, Santhosh and Ravichandran (2012).

The job will be allocated with checkpoint intervals in the standard queue. Save the execution of a job at each period of intervals. Whenever migration takes place, the migrated job will continue its execution from the last saved checkpoint interval, Santhosh and Ravichandran (2013).

Simulation Results:

Our proposed algorithm is simulated with thirty set of jobs based on expected gain, throughput and average waiting time. Two different queues are used and the jobs are executed with manifold resources. We compared our algorithm with non pre-emptive real time scheduling using checkpointing algorithm. This algorithm executes the jobs with a single resource in a non pre-emptive manner through by allocating checkpoint intervals during execution.

The expected gain can be achieved by successful completion of the job within the deadline. Figure 1 shows that our proposed algorithm attains the expected gain when compared with non pre-emptive real time scheduling using checkpointing algorithm.

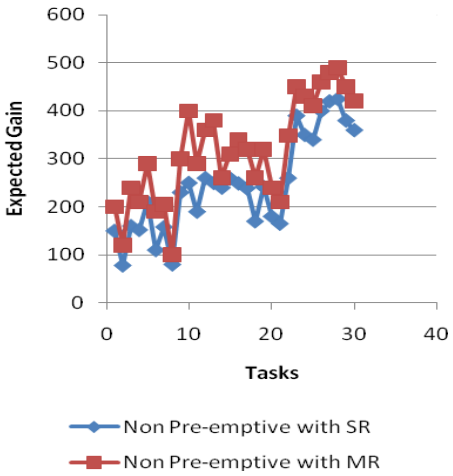

Fig. 1: Comparison of expected gain between Non Pre-emptive with SR and Non Pre- emptive with MR.

The graph for throughput is shown in the figure 2. Non Pre-emptive manifold resources scheduling algorithm shows a better throughput than the non pre-emptive real time scheduling using checkpointing algorithm. Throughput assesses the number of jobs completed within deadline from the group of jobs present in the queue.

Australian Journal of Basic and Applied Sciences, 8(17) November 2014, Pages: 443-446

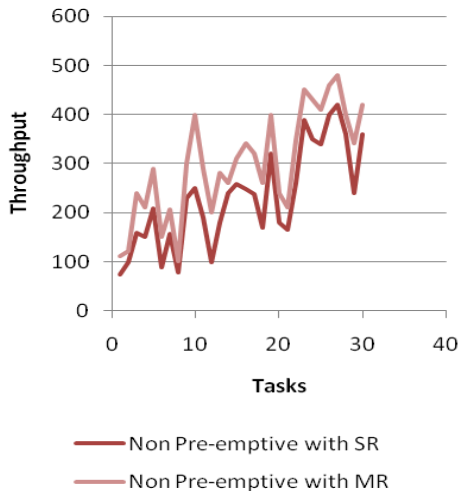

Fig. 2: Comparison of throughput between Non Pre-emptive with SR and Non Pre-emptive with MR.

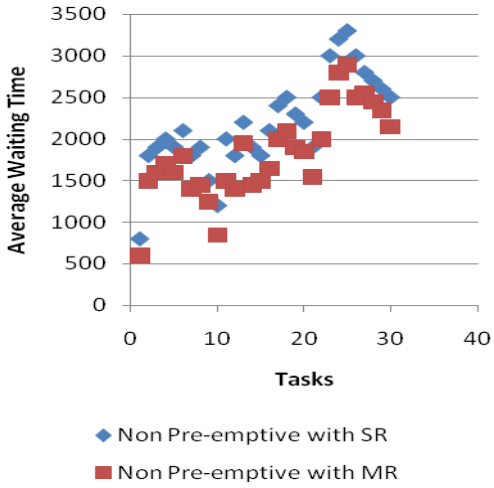

Fig. 3: Comparison of average waiting time between Non Pre-emptive with SR and Non Pre-emptive with MR.

Figure 3 depicts the graph for average waiting time of the job in the queue. In Non Pre-emptive manifold resources scheduling Algorithm, manifold resources are used for processing and executing the jobs. This reduces the average waiting time of the job in the queue when compared to the non pre-emptive real time scheduling using checkpointing algorithm.

Conclusion:

Ever since the cloud computing technology has been introduced, it has a large scope in the field of computation. In this promising field, the number of consumers is growing rapidly in a manner to utilize the services provided by the cloud efficiently. According to that, the cloud providers are extending their services largely over the internet. In the previous model, the service has been provided through a single resource. This increases the average waiting time of the job in the queue. The job has to wait for a longer duration of time. To overcome this problem, we proposed a Non Pre-emptive manifold resources scheduling algorithm. In this algorithm manifold resources are used in order to provide the services quickly. This leads to decrease in the average waiting time of the job in the standard queue. Our simulation result shows better outcomes while compared with the existing algorithm.

REFERENCES

Armbrust, M., A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica and M. Zaharia, 2009. *Above the clouds: A Berkeley view of cloud computing*,UC Berkeley.

Bartal, Y., S. Leonardi, A. Marchetti-Spaccamela, J.S. Gall and L. Stougie, 1996. *Multiprocessor scheduling with rejection*, In Proceedings of SODA, pp: 95-103.

Casati, F. and M. Shan, 2001. *Definition, execution, analysis and optimization of composite e-service*, IEEE Data Engineering.

Idawaty Ahmad, S. Shamala, M. Othman and Muhammad Fauzan Othman, 2008. *A Preemptive Utility Accrual Scheduling Algorithm for Adaptive Real Time System*, IJCSNS International Journal of Computer Science and Network Security, 8(5).

Kuno, H., 2000. *Surveying the e-services technical landscape, In 2nd International Workshop on Advanced Issues of Ecommerce and Web-based Information Systems*.

Li, P., 2004. *Utility Accrual Real-Time Scheduling: Models and Algorithms*, PhD thesis, Virginia Polytechnic Institute and State University.

Li, P., H. Wu, B. Ravindran and E. Jensen, 2006. *A utility accrual scheduling algorithm for real-time activities with mutual exclusion resource constraints, Computers*, IEEE Transactions on, 55(4): 454-469.

Locke, C.D., 1986. *Best-effort decision making for real-time scheduling*, PhD thesis, Carnegie Mellon University.

Mallikarjuna Shastry, P.M., K. Venkatesh, 2010. *Selection of a Checkpoint Interval in Coordinated Checkpointing Protocol for Fault Tolerant Open MPI*,‖ (IJCSE) International Journal on Computer Science and Engineering, 02(06): 2064-2070.

Maria Chtepen, Filip H.A. Claeys, Bart Dhoedt, Filip De Turck, Piet Demeester, 2009*." Adaptive Task Checkpointing And Replication: Toward Efficient Fault-Tolerant Grids*‖, IEEE Transactions On Parallel And Distributed Systems, 20(2).

Mehdi Kargahi, Ali Movaghar, 2006. *A method for performance analysis of Earliest Deadline First Scheduling policy, The Journal of Supercomputing*, 37: 197-222.

Santhosh, R., T. Ravichandran, 2012. *Non-Pre-emptive On-Line Scheduling of Real-Time Services with Task Migration for Cloud Computing*‖, European Journal of Scientific Research ISSN 1450-216X Vol. 89 No 1 October, pp: 163-169.

Santhosh, R., T. Ravichandran, 2013. *Non-Preemptive Real Time scheduling using checkpointing algorithm for cloud computing*‖, International journal of Computer Applications (0975 - 8887), 80(9).

Shou Liu, Gang Quan, Shangping Ren, 2010. *On-line Scheduling of real time services for cloud computing*, In IEEE World congress on services.

Weiss, A., 2007. *Computing in the clouds*, Networker, 11(4): 16-25.

Wu, H., 2005. *Energy-Efficient utility Accrual Real-Time Scheduling*, PhD thesis, Virginia Polytechnic Institute and State University.

Wu, H., B. Ravindran and E.D. Jensen, 2004. *On the joint utility accrual model*, pages 124.

Wu, H., B. Ravindran and E.D. Jensen, 2004. *Utility accrual scheduling under joint utility and resource constraints*, Pages 307.

Wu, H., B. Ravindran and E.D. Jensen, 2010. *Energy-efficient, utility accrual real-time scheduling under the unimodal arbitrary arrival model*, In ACM Design, Automation, and Test in Europe.

Wu, H., U. Balli, B. Ravindran and E. Jensen, 2005. *Utility accrual real-time scheduling under variable cost functions*, pp: 213-219.