World Engineering & Applied Sciences Journal 8 (1): 42-47, 2017
ISSN 2079-2204

DOI: 10.5829/idosi.weasj.2017.42.47

Priority Based Fairshare Scheduling Algorithm in Cloud Computing Environment

A. Durga and R. Madhumathi

Sri Ramakrishna Engineering College, Coimbatore, Tamilnadu, India

Abstract: Cloud computing is the distributed computing, which is the new paradigm but evolving very fast. Middle wares such as SAP/Oracle and business enterprise suits are integrated into the Cloud infrastructure. Due to the large amount of resources available, cloud services are used for multiple science fields like physics, microbiology or weather forecast. In this context, resource utilization should be paid attention in cloud environments. Therefore, the scheduling of the tasks on Cloud environment is the challenging one, without an impact of users and service providers. Fairness in scheduling plays the significant role that could help to enhance the efficiency and provides optimal resource allocation. Most of the organizations have moved on to fair scheduling that it attempts to avoid starvation of resources for longer tasks. There are several scheduling algorithms used in cloud environment. Hadoop by default uses FIFO scheduling. Even though FIFO is the simplest form of scheduling, it has poor fairness. In order to address this problem Priority Based Fairshare Scheduling algorithm is proposed. When Fairshare scheduling is applied, the user who belongs to the highest dynamic priority will be the first job in the queue. The experimental shows that the proposed algorithm achieves good performance in terms of minimizing the execution time and increasing the throughput.

Key word: Cloud computing · Fairness · Fairshare Scheduling · Priority

INTRODUCTION

Cloud Computing has reduced over the period through grid computing, utility computing, virtualization and service oriented architecture. The main advantage of cloud computing is to provide on-demand self-services to the users on pay-per-usage basics i.e. Based on the resources usage on cloud platform the charges are applied, rather than on a flat-rate basis.In order to meet the user needs, Cloud must have a large and flexible resource pool [1]. It contains the resource, which is formed by interconnecting the various computing resources and virtual computing resources [2]. There are many popular cloud computing environments, such as Amazon EC2 [3] and IBM Smart Cloud [4].It offers the services through the internet anytime, anywhere to the end users. The services are provisioned in the form of computing, storage and platform for developing the software. Cloud is divided in to Infrastructure-as-a-Service (IaaS), Platform-as-a Service (PaaS) and Software-as-a-Service (SaaS) [5, 6].

Important feature of cloud computing is Virtualization. Virtualization is a technology which supports the physical resources shared by the logical independent multi-applications located in the same computer node. In order to improve the node utilization, virtualization provides a feasible solution. As the demand is being increased for the virtual machines (VMs), the system configuration methodology as automated through computing on most of the datacenter. As an outcome, it remains difficult to guarantee the resource utilization by using virtualization technology [6–8], as it is difficult to extent it could only suitable for the large scale groups which allows providing multiple VMs on a single physical machine. It mainly used for improving the resource utilization, reducing the costs and easing the server management [9, 10].

Applications have variable and dynamic needs, in which capacity management and demand prediction are especially a complicated one. Cloud resource scheduling has been one of the key challenges in cloud computing. Scheduling plays a vital role in order to achieve the high performance for running real applications and best system throughput and it is mapped to machines so that deadlines and response time are satisfied. Two objectives in cloud resource scheduling are 1. To maximize the provider's expected profit, 2.To maximally satisfy each user's resource requirements. Schedulers assign resources to tasks (or tasks to resources) at specific time.

Correponding Author: A. Durga, Sri Ramakrishna Engineering College, Coimbatore, Tamilnadu, India.

Fairness in scheduling is an important criterion that improves the efficiency and also provides optimal resource allocation. Fair share scheduling is a technique that ensures fairness, faster response time and minimum delay. It also ensures resource allocation in a fair manner and is an added advantage. One of the major issues concerned with scheduling is that it possesses higher overhead while scheduling more number of tasks. Request partitioning/ request scheduling is one way of reducing the overhead while scheduling [11].while making scheduling decision, Fair share scheduler considers the execution history of a related group of processes, along with the individual execution history of each process The user community is divided into a fair- share groups. A fraction of CPU is allocated to each group. So that it attempts to avoid starvation of resources for longer tasks.

This paper introduces Priority based Fair Share Scheduling. This algorithm has three parts 1.Dispatching the task into queue 2.Allocating the Shares to the task by Fair Share Scheduler. 3. Calculating the priority for the task. Then according to the priority the task are scheduled.

In the experiments Hybrid Scheduling Algorithm Based on Resource Clustering is compared with the proposed algorithm. The experiment results show that the proposed PFSS algorithm can achieve good performance in terms of minimizing the execution time and increasing the throughput.

The rest of the paper is organized as follows. Section II presents the related work. In Section III it describes the resource allocation problem and Section IV describes the proposed Priority based Fairshare Scheduling algorithm. Section V presents the experimental results. Section VI concludes the paper and discusses some future work.

Related Works: Scheduling in a cloud environment is challenging due to dynamic and Heterogeneous resources spread over geographical area.

Z. Peng *et al.* [12] proposed the scheduling scheme based on Reinforcement Learning. It mainly focused on efficient task scheduling under resource constraints problems by introducing fine grained system model in cloud computing and optimization task scheduling scheme. The proposed model consists of separate sub models including task schedule submodel, task execute submodel and task transmission submodel. By these submodel it can accurately analyzed the user request. By using queuing theory the performance analysis of response time is obtained. User task scheduling is achieved by using reinforcement learning strategies. It considers both the resources adaptation adjustment to workload and system dynamics.

The concept of hybrid approach for tasks scheduling in Heterogeneous distributed systems is described in [13] by Mihaela-Andreea Vasile *et al.* This approach used for different types of application such as batch jobs and workflows. Tasks are executed in two phases, tasks are assigned to groups of resources in the first phase and in the second phase and different scheduling algorithm is used for each group of resources. Scheduling strategy is based on using different scheduling strategies, which considers both the heterogeneity of computing resources and application tasks and flows. The effectiveness of the utilization of the computing resources is determined by the efficiency of the allocation strategy of the resources.

Chun-Wei Tsai *et al.* [14] introduces the concept of Hyper-heuristics scheduling. The main aim is to find the better scheduling solutions for cloud computing systems. Proposed algorithm is used to dynamically determine which low-level heuristic is to be used in finding better candidate solution by using operators called diversity detection and improvement detection. This algorithm can be applied for both Sequence-dependent and Sequence-independent scheduling problems. It reduces the makespan of task scheduling.

Antony Thomasa *et al.* [15] proposed credit based scheduling algorithm. It uses two parameter 1.Task Length 2.User Priority. This algorithm is mainly focus on credit system. For each task, a credit based on their task length and priority is assigned. In the scheduling of the task, these credits will be considered different tasks are assigned different credits. Based on the priority which is assigned to the each task credit value is generated. Results show a considerable improvement in the utilization of resources.

Zhanjie Wang *et al.* [16] proposed dynamically hierarchical resource-allocation. This algorithm is proposed for multiple cloud nodes collaborating in big data environment. The algorithm dynamically divides tasks and nodes into different levels based on computing power and storage factors by using fuzzy pattern recognition. In between tasks and nodes, a dynamically adjusted mapping will be generated. In case of arriving the new task, only the nodes corresponding to the task level join in the bid. This algorithm reduces the communication traffic during resource allocation. Both theory and experimental results shows that the proposed algorithm has advantage of decreasing the communication traffic and increasing the makespan.

R.N. Calheiros *et al.* [17] discussed about Cloudsim. Cloudsim is the most used Cloud simulator. It consists of default scheduling policies available for In CloudSim there are available defaults scheduling policies both for VMs allocation on hosts and for tasks allocation. The simulator

offers space-shared and time-shared policies for VMs and tasks provision and those two available policies may be used in every different effect in tasks execution. iCanCloud,which is a similar tool. it is a flexible and scalable Cloud infrastructure simulator [18].

Problem Formulation: There has been rapid growth in Cloud computing systems it allows for the provision of resources on-demand. There are different challenges; cloud providers want to maximize revenue by achieving high resource utilization. In order to minimize the expenses user wants to meet their performance requirements. Several problems with the process of resource allocation remain unaddressed. The proposed work describes a resource allocation scheme that achieves high performance and fairness in a cloud computing environment.

Proposed Work: There are several scheduling algorithms used in cloud environment. Hadoop by default uses FIFO scheduling. Even though FIFO is the simplest form of scheduling, it has poor fairness. It is the reason that most of the organizations have moved on to fair scheduling that it attempts to avoid starvation of resources for longer tasks. For example Linux kernel is found to use a Completely Fair Scheduling algorithm [19]. Priority plays a major role in determining fairness. Scheduling based on priority maximizes the profit.

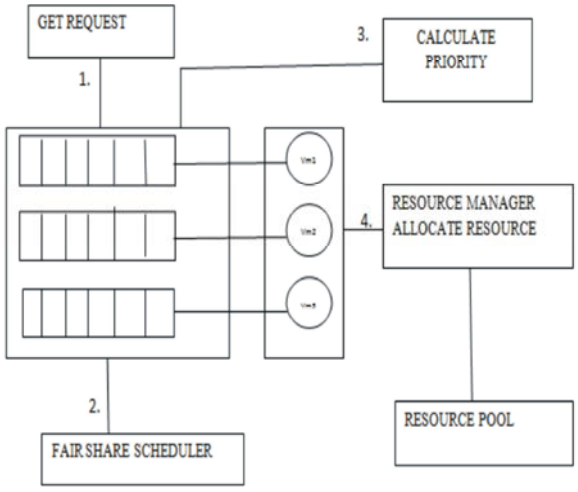

Fig. 1: Archirecture of the Proposed Sytsem

The proposed system consists of global queues, fair share scheduler, Scheduler. Queues are used to store the user request. Fairshare Scheduler is used to allocate the share. Scheduler is used to calculate the priority. Incoming User request are dispatched to the queues according to the memory.

The main aim of this scheduler is to allocate the shares to each users or group, the work of fairshare policy is to define the order so that scheduler attempts to place jobs that are in a queue. Shares are assigned to the users or groups by using the syntax,

[Users, number_shares]

Fairshare policy allocates fixed number of shares to each user in the queue. The Shares will be allocated based on the available resources. It represents the fraction of the resources. When Fairshare scheduling is applied, the user who belongs to the highest dynamic priority will be the first job in the queue. It doesn't consider the order of the job as main objective. It focuses on the user who has the highest dynamic priority. If the user has no shares it cannot run the job. When fairshare scheduling is used, scheduler tries to place the first job in the queue that belongs to the user with the highest dynamic priority. The most important thing is the dynamic priority of the user who submitted the job.

ALGORITHM-I FOR FAIRSHARE SCHEDULER

Step 1: Get the request from the user
Step 2: Dispatch the request to the different queues
Step 3: Allocate the shares to the queues based on the available memory
Step 3.1: calculate the total memory in the queue (Total Memory)
Step 3.2: Shares are allocated to the queue by using, Total share = (Available Memory/Total Memory) * 100 %
Step 4: Then the shares are divided and allocate to the individual request
Step 5: Priority are calculated for each request
Step 6: Sort the user with respect to priority for each queue
Step 6: picks the task with highest priority in the queue (i.e. the job with highest priority runs first)
Step 7: After the completion of the task repeat the step 6 until queue is empty.

User Priority: By default, scheduler calculates the priority for each user based on:

- The number of shares assigned to the user
- The resources used by jobs belonging to the user: a. Number of job slots reserved and in use, b. Run time of running jobs, c.CPU time used by the job.

For queue-level fairshare, scheduler measures the resource consumption of all the user's jobs in the queue. This means a user's dynamic priority can be different in every queue.

Priority Formula: By default, scheduler calculates priority according to the following formula [20]:

Priority = number_shares / (CPU time * CPU timefactor + runtime * runtime factor + (1 + job slots) * run job factor)

By default CPU time weighting factor and runtime factor is 0.7, job slots weighting factor is 3.

After the scheduler calculates the priority the task with the highest priority runs first. (I.e. the important user is scheduled first)

ALGORITHM-II PRIORITY BASED FAIRSHARE SCHEDULING

Step 1: Get the request from the user
Step 2: Dispatch the request to the queue (q_1, q_2, q_3...)
Step 3: Allocate the shares to the request by using algorithm I
Step 4: Calculate the priority according to the formula
Step 5: Sort the request according to the priority
Step 6: picks the task with highest priority
Step 7: Repeat the step 6 until the queue is empty.

RESULTS AND DISCUSSIIONS

The experimental tests were conducted among existing and proposed methodology with varying number of tasks. The comparison is made between the proposed methodology and the Hybrid Scheduling Algorithm Based on Resource Clustering.The performance metrics considered for comparing the existing and proposed approaches are average execution time, throughput. For simulation cloudsim toolkit is used to evaluate the results.

Average Execution Time: The total time taken to process the submitted task is defined as the execution time. The execute on time of the proposed methodology need to be reduced in order to improve the system performance

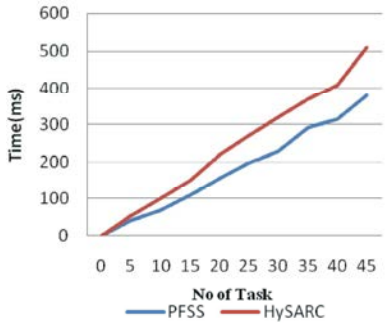

Fig. 2: Comparison of Average Execution Time

From the fig: 2, it is observed that the proposed methodology can complete the task execution with less execution time than the existing method. In this graph, x axis plots the total number of tasks and y axis plots the execution time in milli second.

Throughput: Throughput is defined as the total number of tasks that can be executed in a particular period of time. High throughput of the system defines the better performance of the system [21].

Throughput = (number of tasks completed / Total number of task) per minute

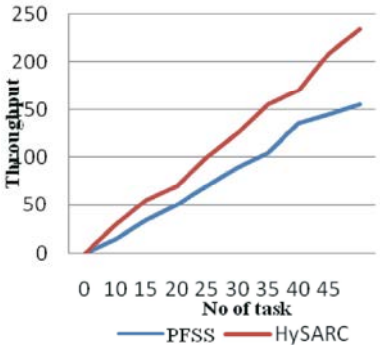

Fig. 3: Comparison of Throughput

From Fig: 3 it is observed that the proposed methodology can complete the task execution with more throughput than the existing method. In this graph, x axis plots the total number of tasks and y axis plots the throughput.

CONCLUSION

Priority based Fairshare scheduling algorithm is proposed for avoiding starvation of resources for longer tasks. It is evaluated on the cloudsim environment,

compared it with the Shortest Job First and Resource aware hybrid scheduling algorithm. This algorithm considers both the fairness and priority. Shares are allocated to the users based on available memory by Fairshare scheduler. The task with highest priority will schedule first (i.e. the important users) by using the Priority based Fairshare Scheduling Algorithm. This algorithm achieves Fairness and improves the efficiency and provides optimal resource allocation. The experiment result demonstrates this algorithm achieves good performance in terms of minimizing the execution time and increasing the throughput.

This algorithm focuses on the used job's slot. In the future work priority will consider unused job slots effectively blocked by another job. Also aim at implementing and validating our strategies in a real-world cloud computing environment.

REFERENCE

1. Marinescu, D.C., 2013. Cloud Computing: Theory and Practice, first ed., Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, ISBN: 0124046274, 9780124046276.
2. Buyya, R., C.S. Yeo, S. Venugopal, J. Broberg and I. Brandic, 2009. Cloud computing and emerging ITplatforms: vision, hype and reality for delivering computing as the 5th utility. Future Gener ComputSyst, 25: 599-616.
3. Amazon, E.C., 2015. 2 http://aws.amazon.com/cn/ec2/. Accessed 7 Feb 2015.
4. IBM SmartCloud, 2015. http://www.ibm.com/cloud-computing/cn/zh/index.html. Accessed 7 vFeb2015
5. Zhou, M., R. Zhang and D. Zeng, 2010. Services in the cloud computing er a: a survey. In: Proceeding of the Fourth InternationalUniversal Communication Symposium (FIUCS2010), pp: 40-46.
6. Buyya, R., C. Yeo and S. Venugopal, 2008. Market-oriented cloud computing:vision, hype and reality for delivering it services as computing utilities. In: Proceeding of the 10th IEEE International Conference on High Performance Computing and Communications (HPCC2008), pp: 5-13.
7. Karve, A., T. Kimbrel and G. Pacifici, 2006. Dynamic placement for clusteredWeb applications, In: Proceeding of the 15th International Conference on WorldWideweb (WWW2006), pp: 593-604.
8. Heath, T., B. Diniz and E.V. Carrera, 2005. Energy conservation in heterogeneous server clusters. In: Proceeding of the 10th ACMSIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP2005), pp: 186-195.
9. Barham, P., B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho and A. Warfield, 2003. Xen and the art of virtualization, Proc ACM SIGOPS Oper Syst Rev, 37(5): 164-177.
10. Begnum, K., 2012. simplified cloud-oriented virtual machine management with MLN. J Supercomputer, 61(2): 251-266.
11. Avi-Itzhak, B., H. Levy and D. Raz, 2007. A resource allocation queueing fairness measure: Properties and bounds, Queueing Systems, 56(2): 65-71.
12. Peng, Z., D. Cui, J. Zuo, Q. Li, B. Xu and W. Lin, 2015. Random task scheduling scheme based on reinforcement learning in cloud computing. Cluster Computing, 18(4): 1595-1607.
13. Mihaela-Andreea Vasile A, Florin Popa, Radu-Ioan Tutueanua, Valentin Cristea A and Joanna Ko³odziej, 2015. Resource-aware hybrid scheduling algorithm in heterogeneous distributed computing, Future Generation Computer Systems, 51: 61-71.
14. Tsai Chun-Wei, Wei-Cheng Huang, Meng-Hsiu Chiang, Ming-Chao Chiang and Chu-Sing Yang, 2014. A Hyper-Heuristic Scheduling Algorithm for Cloud, IEEE Transactions on Cloud Computing, 2(2).
15. Thomasa Antony, Krishnalal Ga and V P Jagathy Raj, 2015. Credit Based Scheduling Algorithm in Cloud Computing Environment, Procedia Computer Science, 46: 913-920.
16. Wang Zhanjie and Xianxian Su, 2015. Dynamically hierarchical resource-allocation algorithm in cloud computing environment, J. Supercomput, 71: 2748-2766.
17. Calheiros, R.N., R. Ranjan, A. Beloglazov, C.A.F. De Rose and R. Buyya, 2011. CloudSim:a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, Softw. Pract. Exp. (ISSN: 0038-0644), 41(1): 23-50.
18. Núñez, A., J.L. Vázquez-Poletti, A.C. Caminero, G.G. Castane, J. Carretero and I.M. Llorente, 2012. iCanCloud: a flexible and scalable cloud infrastructure simulator,J. Grid Comput. (ISSN: 1570-7873), 10(1): 185-209.

19. Anuradha, B. and S. Rajasulochana, 2013. Fairness As Justice Evaluator In Scheduling Cloud Resources: A Survey, 3(2): 91-99
20. IBM https://www.ibm.com/support/knowledgecenter/SSETD4_9.1.3/lsf_ae_using/dynamic_user_priority_gb.dita
21. https://docs.oracle.com/cd/E1904401/sol.contaners/817-1592/rmfss-1/index.html