Research Journal of Recent Sciences _____ ISSN 2277-2502
Vol. 4**(IVC-2015),** 29-33 **(2015)** *Res. J. Recent. Sci.*

Token-based Predictive Scheduling of Tasks in Cloud Data-centers

Kumar Narander and Saxena Swati
Department of Computer Science, B. B. A. University (A Central University), Lucknow, UP, INDIA

Available online at: www.isca.in, www.isca.me
Received 4th April 2015, revised 7th May 2015, accepted 29th May 2015

Abstract

Resource Management in a utility-based system such as cloud computing requires a careful observation of users demands and availability of resources. For optimal resource provisioning, an effective task/job scheduling is required which must guarantee fair chance to users and profit to service providers along with maximum utilization of resources. This paper presents a token-based scheduling mechanism which lines up tasks to resources in a fair manner based on a user's token value. A user's token is characterized by his/her SLA parameters, his/her duration in the task queue and the task's nature, i.e. computation-intensive, memory-intensive or communication-intensive. Further, to ensure optimal usage to cloud's resources, the token-based scheduling is complemented by a predictive scheduling which matches user's demands with resource's supply, and delays a task in case it's demand is not currently fulfilled by a machine by giving preference to another task. The experimental results of the proposed work strengthen our claim of fairness, profitability and effective resource management.

Keywords: Cloud datacenters, task scheduling, token, profitability, resource management.

Introduction

Task scheduling in a distributed environment, like cloud computing, is an important factor which influences the performance of these systems. Due to random and parallel applications, it becomes crucial to distribute idle resources among demanding tasks in an effective and fair manner. User's preferences and application's requirements must be taken into account for its effective scheduling. Most of the scheduling policies implemented in the present day give very limited credit to a user's and his application's requirements. However, in reality, different types of applications have different desirable requirements. For example, an intensive computational task requires equal distribution of resources among users, whereas, an interactive application focuses on available resources which are needed at a particular time. In a typical scenario, when a user's application/task is submitted to a cloud system, its attributes are also submitted which help in proper scheduling of the task. These attributes include individual resource requirements (like CPU, bandwidth and memory requirements), response time/deadline requirement and resources redemption preferences among others. Based on these requirements or preferences, a task is allocated to a server or physical machine. An individual physical machine will, in turn, schedule its tasks in an optimized manner which ensures proper utilization of its resources.

This paper considers a cloud datacenter with heterogeneous resources. A number of incoming user jobs are competing for these resources in varied quantities. An ideal solution must service all these concurrent jobs while maintaining quality of service parameters and ensuring optimum usage of cloud's resources. However, continuously increasing user requirements and limited quantity of resources often makes it difficult to satisfy every demand without compromising performance. Performance parameters of any scheduling algorithm include turnaround time, waiting time and response time. This paper is an attempt to devise a scheduling technique which not only ensures low turnaround time and waiting time when compared to FCFS schedule but also this technique guarantees fairness to cloud users as it balances the demand curve with allocation frequency.

Methodology

Related work: Cloud computing is a computing environment which entails huge amount of data processing using various resources in 'pay as you use' manner. The ever increasing number of cloud jobs in a cloud data-center leads to uneven and unfair utilization of cloud resources, resulting in escalating processing costs and increasing carbon emission. To control this scenario, researchers have proposed job scheduling among resources as an effective tool to efficiently utilize various resources.

To curb carbon footprints, a decision framework is developed which monitors the energy consumption of both local and remote sites and accordingly decides to transfer data between them[1]. This movement of data from one site to another, however, will increase the transmission overhead; hence, the proposed idea does not give a very favorable result. Another approach to manage cloud resources is based on personalized user requirements and resources runtime behavior[2]. This approach requires full knowledge of incoming jobs which is highly unlikely in a cloud environment. A comprehensive study on various job scheduling techniques focuses on various

performance parameters such as load-balancing, resource management, response time, etc[3]. Cache based analytical model of resource management in cloud datacenters tries to study the incoming user requests pattern and uses cache nodes to fulfill the requests of identifiable jobs[4]. However, the feasibility of recognizing user patterns in a diverse environment like cloud computing still needs to be proven. Among all the processing data, research data is of prime concern and requires focus on its storage and issue concerning its preservation[5]. In the recent past, context aware computing is focused to improve resource utilization in cloud systems and based on this computing, an adaptive job scheduling is proposed to improve QoS parameters[6]. Traditional scheduling methods, such as FCFS and Round-Robin, are compared with a priority-based scheduling technique which is characterized by the dynamic nature of a cloud job[7]. Factors effecting the energy consumption in a cloud data-center are analyzed in order to increase energy efficiency without adversely affecting its performance and QoS parameters[8]. With the aim to reduce computational cost of data-intensive applications, various evolutionary and swarm-based job scheduling techniques are discussed[9]. Multi-objective based scheduling methods are proposed wherein job deadline is given a priority[9,10]. Techniques of co-scheduling like predictive and proportional-sharing are summarized and their variants are also highlighted with respect to distributed computing[11]. It will be interesting to watch their application in cloud computing, in particular. On the other hand, dynamic programming based scheduling technique is presented for heterogeneous jobs with varied execution and release time[12].

So far in the literature review, emphasis was given to scheduling jobs for efficient resource management. A new insight is given which deals with virtual machine pre-emption, so as to accommodate several concurrent jobs in a cloud environment[13]. This technique, however, require comprehensive tracing and synchronization mechanisms, further increasing the workload of a scheduler. Efficient power consumption is the main objective of task scheduling and genetic algorithm is used based on MapReduce[14,15,16]. Similarly genetic algorithm along with artificial neural networks is used to layout a scheduling task to utilize cloud resources in an optimal manner[17]. Scheduling of both sequential and parallel tasks in grid/distributed scenarios is given with backfilling to add an error margin for a task's execution-time error calculation[18]. A performance comparison of stride scheduling with lottery scheduling in distributed networks is reviewed along with their various types[19,20]. A detailed study of algorithms and scheduling techniques used in cloud computing is given with their challenges and benefits[21].

Most of the work done in job scheduling revolves around reducing energy consumption in a cloud datacenter or reduction in operational cost which leaves a lot of scope for further improvement. The basic idea of scheduling, however, is to effectively reduce operational time while servicing most of the incoming requests. This paper proposes a token based predictive scheduling scheme which promises a reduction in operational time by lowering turnaround time and waiting time.

Proposed Scheduling Technique: The proposed token-based predictive scheduling is based on the relative resource requirements of concurrent jobs in a cloud environment. Resources considered in this paper are central processing units, memory and network bandwidth. Tokens are entities that reflect the heterogeneous demands of jobs for the above mentioned cloud resources. Based on these tokens, a schedule is sketched out which keeps the job with highest token in the first place and so on. Allocation interval for each job is fixed in such a manner that the allocation interval of a job is inversely proportional to the schedule sequence. Allocation interval defines the time gap between successive allocations of resources to a job. For example, suppose the token values of jobs P_i and P_q are T_i and T_q respectively, such that $T_q > T_i$, this conveys two things: i. The resource requirement of job P_q is more than P_i and ii. Allocation interval $A_q < A_i$, i.e., job P_q will be frequently allotted the required resources as compared to P_i.

Thereafter, allocation duration for the schedule is calculated. Presently in this paper, two variants of the schedule are created- schedule1 where allocation duration is the quotient of division of the highest and the lowest resource requirement's sum by the total number of concurrent jobs, whereas in schedule2, allocation duration is fixed as one time unit.

As an example of the proposed schedule, consider 5 concurrent jobs P_1, P_2, P_3, P_4 and P_5 with their individual resource requirements as follows in figure-1.

Resources/Jobs	P_1	P_2	P_3	P_4	P_5
CPU	3	2	5	1	7
Memory	4	6	3	2	8
N/W bandwidth	5	8	9	6	1

Figure-1
Token scheduling example

Based on these requirements, token for each job and for each resource is set by comparing the individual resource requirements for each resource. In figure-1, consider the resource CPU wherein job P_5 has the highest requirement and P_4 has the least. Hence, P_5 token value for CPU will be the highest and P_4 token value will be the least. CPU token for a job P_i is designated as C_i, memory token as M_i and bandwidth token as B_i. Therefore, as per figure-1, these tokens are arranged in descending order for each resource, i.e.

$C_5 > C_3 > C_1 > C_2 > C_4$
$M_5 > M_2 > M_1 > M_3 > M_4$
$B_3 > B_2 > B_4 > B_1 > B_5$

Sequence	P_5 -> P_2 -> P_1 -> P_3 - > P_4
Allocation Interval	1 < 2 < 3 < 4 < 5
Allocation Duration (schedule2)	3 (schedule1) and 1

Figure-2
Token-based sequence, allocation interval and duration

Now, a job sequence is formed based on the above sequence of tokens. This sequence compares the token sequence of three resources and picks a job that dominates in its token value. For example, among the highest tokens C_5, M_5 and B_3 it is clear that job P_5 has a majority of high tokens, hence P_5 will be put first in sequence. Similarly, among the second highest tokens C_3, M_2 and B_2, job P_2 wins the majority, so P_5 is followed by P_2 in the sequence. Based on this sequence, allocation interval for each job is fixed in such a way that the former jobs get a lesser interval than the latter ones. Allocation interval defines the frequency of resource allocation to a job. Thus, P_5 will have the least allocation interval and P_4 will have the highest. The allocation interval of each job based on the sequence is shown in figure-2. Two variants of the above sequence are formed- schedule1 and schedule2. Schedule1 has an allocation duration of 3 time units which is obtained by dividing the sum of the highest and lowest resource requirements by the number of jobs. Schedule2 keeps fixed allocation duration of 1 time units. Figure-3 shows both schedule1 and schedule2.

Results and Discussion

The proposed scheduling scheme showcases two schedules- Schedule1 and Schedule2. Schedule1 has a fixed allocation duration of 3 time units with variable allocation interval based on job token values. On the other hand, schedule2 has the same allocation interval as schedule1 with allocation duration of a single time unit. Both the schedules are based on token-based predictive job scheduling technique and are compared with FCFS scheduling mechanism as shown in figure-4. Here, grey areas depict the completion of a job.

Figure-5 and figure-6 compare the turnaround time of schedule1, schedule2 and FCFS with five concurrent jobs P_1 to P_5 having different token values as given in the example above. Figure-5 compares the turnaround time between the two schedules of the proposed schedule scheme whereas figure-6 compares the turnaround time among both the schedules and FCFS.

Figure-3
Schedule1 and Schedule2

Figure-4
Schedule 1, schedule 2 of token-based scheduling and FCFS schedule

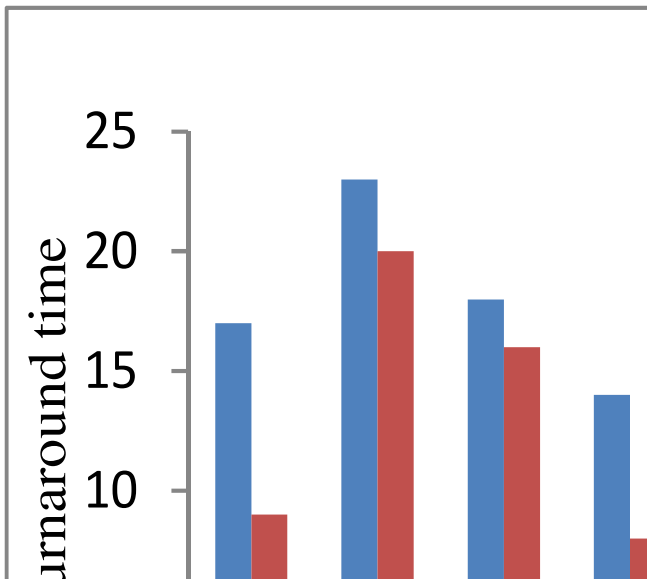

Figure-5
Turnaround time in Schedule1 and 2

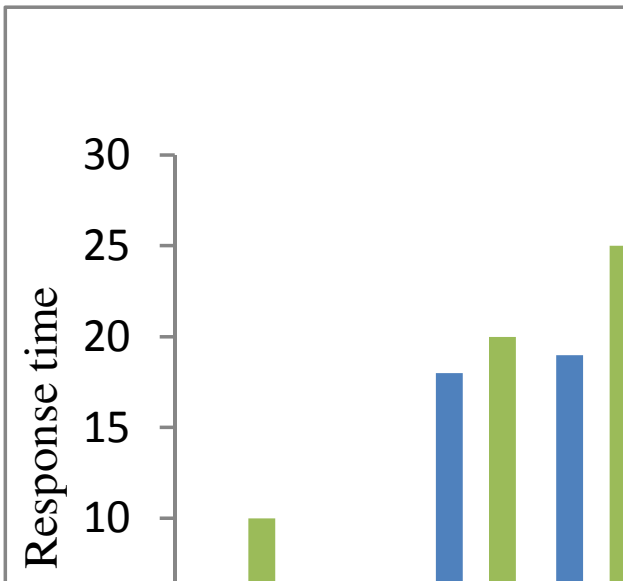

Figure-7
Response time comparison

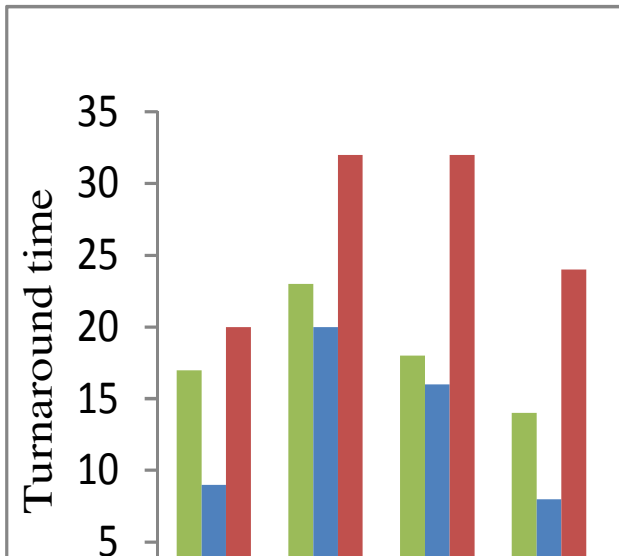

Figure-6
Comparison of turnaround times

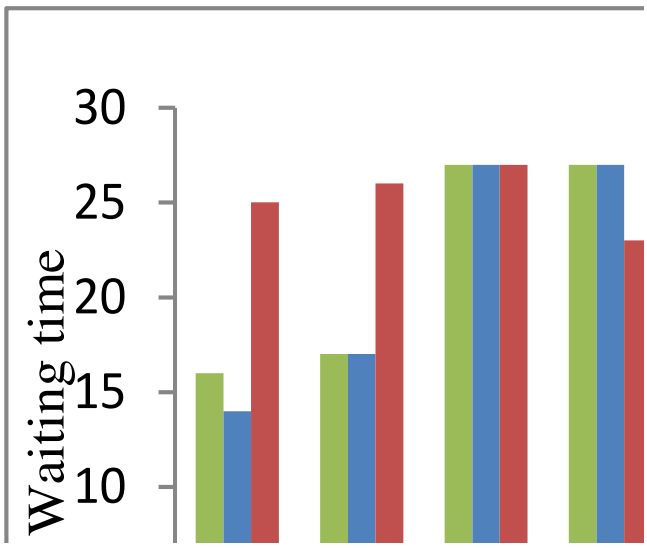

Figure-8
Waiting time comparison

Figure-7 compares the variants of token-based predictive scheduling and FCFS scheduling with respect to the response times of concurrent jobs. Response time is the time from submission of a job to the first time it is scheduled.

Figure-8 compares the three scheduling scenarios in their waiting time. Here, waiting time is the time spends by a job in the ready queue. As the figures show, the presented scheduling scheme fares reasonably well as compared to the traditional FCFS scheme with respect to turnaround time and waiting time.

Conclusion

Job scheduling is an integral part of a datacenter's networking. An effective job scheduling should be fair to cloud users and at the same time it must utilize a datacenter's resources in the most optimized way. This paper proposes a token-based job scheduling mechanism along with its variation which ensures low waiting time and low response time. In case of conflicting jobs, i.e. jobs with same requirements, a prediction-based technique is added to the proposed token-based scheduling which judicially chooses one job over another by monitoring their past track records and predicting their future demands. The proposed scheme and its variation are compared with FCFS

scheduling mechanism and a lower turnaround time and wait time is noted.

As a future work, further enhancements like resource usage tracking and virtual machine migration reduction can be added to this scheduling so as to further improve a datacenter's performance.

References

1. Makkes M.X., Taal A., Osseyran A. and Grosso P., A decision framework for placement of applications in clouds that minimizes their carbon footprint., *Journal of Cloud Computing*, **2(1), 1-13, (2013)**

2. Chen X., Zhang Y., Huang G., Zheng X., Guo W. and Rong C., Architecture-based integrated management of diverse cloud resources, *Journal of Cloud Computing*, **3(1), 1-15, (2014)**

3. Pinal Salot, A Survey of Various Scheduling Algorithm in Cloud Computing Environment, *International Journal of Research in Engineering and Technology*, **2(2), (2013)**

4. Sithole E., McConnell A., McClean S., Parr G., Scotney B., Moore A. and Bustard D, Cache performance models for quality of service compliance in storage clouds, *Journal of Cloud Computing*, **2(1), 1-24, (2013)**

5. Waddington S., Zhang J., Knight G., Jensen J., Downing R. and Ketley C, Cloud repositories for research data–addressing the needs of researchers, *Journal of Cloud Computing*, **2(1), 1-27, (2013)**

6. Assunçao M.D., Netto M.A., Koch F. and Bianchi S., Context-aware job scheduling for cloud computing environments, In *Proceedings of the 2012 IEEE/ACM Fifth International Conference on Utility and Cloud Computing* **,** 255-262, **(2012)**

7. Agarwal D. and Jain S., Efficient optimal algorithm of task scheduling in cloud computing environment, *arXiv preprint arXiv,* **140,** 2076, **(2014)**

8. Banerjee A., Agrawal P. and Iyengar N.C.S., Energy Efficiency Model for Cloud Computing. *International Journal of Energy, Information and Communications,* **4,** 29-42, **(2013)**

9. Bilgaiyan S., Sagnika S. and Das M., An Analysis of Task Scheduling in Cloud Computing using Evolutionary and Swarm-based Algorithms, *International Journal of Computer Applications*, **89(2),** 11-18, **(2014)**

10. Liu J., Luo X.G., Zhang X. M., Zhang F. and Li B.N., Job scheduling model for cloud computing based on multi-objective genetic algorithm, *IJCSI International Journal of Computer Science Issues*, **10(1),** 134-139**, (2013)**

11. Liang D., Ho P.J. and Liu B., Scheduling in Distributed Systems, **(2000)**

12. Chang H.J., Wu J.J. and Liu P., Job scheduling techniques for distributed systems with heterogeneous processor cardinality, In *Pervasive Systems, Algorithms, and Networks (ISPAN), 2009 10th International Symposium on,* 57-62**, (2009)**

13. Gebai M., Giraldeau F. and Dagenais M.R., Fine-grained preemption analysis for latency investigation across virtual machines, *Journal of Cloud Computing: Advances, Systems and Applications*, **3(1),** 41, **(2014)**

14. Ghanbari S. and Othman M., A priority based job scheduling algorithm in cloud computing, *Procedia Engineering*, **50**, 778-785, **(2012)**

15. Luo L., Wu W., Di D., Zhang F., Yan Y. and Mao Y., A resource scheduling algorithm of cloud computing based on energy efficient optimization methods, In *2012 International Green Computing Conference (IGCC),* 1-6, **(2012)**

16. Wang X., Wang Y. and Zhu H., Energy-efficient task scheduling model based on MapReduce for cloud computing using genetic algorithm, *Journal of Computers*, *7***(12),** 2962-2970**, (2012)**

17. Maqableh M., Karajeh H. and Masa'deh R.E., Job Scheduling for Cloud Computing Using Neural Networks, *Communications and Network*, **(2014)**

18. Dimitriadou S.K. and Karatza H.D., Job scheduling in a distributed system using backfilling with inaccurate runtime computations, In *Complex, Intelligent and Software Intensive Systems (CISIS), 2010 International Conference on*, 329-336, **(2010)**

19. Waldspurger C.A. and Weihl W.E., Stride scheduling: Deterministic proportional share resource managemen,. *Massachusetts Institute of Technology, Laboratory for Computer Science,* **(1995)**

20. Waldspurger C.A. and Weihl W.E., Lottery scheduling: Flexible proportional-share resource management, In *Proceedings of the 1st USENIX conference on Operating Systems Design and Implementation* (p. 1). USENIX Association, **(1994)**

21. Chawla Y. and Bhonsle M., A Study on Scheduling Methods in Cloud Computing, *International Journal of Emerging Trends and Technology in Computer Science (IJETTCS)*, **1(3),** 12-17, **(2012)**