Asian Journal of Applied Science and Technology (AJAST)
(Open Access Quarterly International Journal) Volume 2, Issue 2, Pages 821-826, April-June 2018

Multiple Virtual Machine In A Cumulative Server Using Xen Balloon Driver By Implementing Real Time Task Oriented Task Scheduling Algorithm

S.Devarajsamy[1] and R.Sudhakar[2]

[1]*PG Student, Department of CSE, Nandha College of Technology. India. Email: kdrajusss@gmail.com*
[2]*Assistant Professor, Department of CSE, Nandha College of Technology, India. Email: sudhakarcs87@gmail.com*

Article Received: 01 March 2018 *Article Accepted: 09 April 2018* *Article Published: 28 April 2018*

ABSTRACT

A system for automatic memory control based on the balloon driver in VMs. Researchers can download our toolkit. The project aims to optimize the running times of applications in consolidated environments by overbooking and/or balancing the memory pages of VMs. The system is lightweight and can be completely integrated into user space without interfering with VMM operation. Design a global-scheduling algorithm based on the dynamic baseline to determine the optimal allocation of memory globally. In this existing system evaluate optimized solution to memory allocation using real workloads that run across VMs. The virtualization technique enables multiple virtual machines (VMs) to be placed on the same physical hosts and supports the live migration of VMs between physical hosts based on the performance requirements. When VMs do not use all the provided resources, they can be logically resized and consolidated to the minimum number of physical hosts, while idle nodes can be switched to sleep or hibernate mode to eliminate the idle energy consumption and thus reduce the total energy consumption in cloud data centers. Cloud can achieve the same level of computing power as a supercomputer does, but at a much reduced cost. Cloud is like a virtual supercomputer. However, need to consider about many conditions such as network status and resource status because the members of Cloud are connected by networks. Cloud is also a heterogeneous system. Scheduling independent tasks on it is more complicated. In order to utilize the power of Cloud computing completely, need an efficient task scheduling algorithm to assign tasks to resources. This paper focuses on the efficient task scheduling EASJSA considering the completion time of tasks in a cloud environment.

Keywords: Cloud, Resource Scheduling, TEC, EASJSA, Energy Consumption, MEB.

1. INTRODUCTION

The cloud computing is the phrase used to describe different scheme in which computing resource is delivered as a service over a network connection. Cloud computing is a type of computing that relies on sharing a pool of physical and/or virtual resources, rather than deploying local or personal hardware and software Cloud can achieve the same level of computing power as a supercomputer does, but at a much reduced cost.

Cloud is like a virtual supercomputer. However, we need to consider about many conditions such as network status and resource status because the members of Cloud are connected by networks. Cloud is also a heterogeneous system. Organize independent tasks are more complicated. In order to utilize the power of Cloud computing, we need a dynamic job scheduling algorithm to assign jobs to resources. This paper focuses on the efficient job scheduling considering the completion time of jobs in a Cloud environment.

The paper considers Adaptive Scoring Job Scheduling algorithm EASJS which aims to decrease job's completion time. We consider not only the computing power of each resource in the Cloud but also the transmission power of each cluster in a Cloud system. The computing power of each resource is defined as the product of CPU speed and available CPU percentage and the transmission power of each cluster is defined as the average bandwidth between different clusters. EASJS uses the status of each resource in the Cloud as parameters to initialize the cluster score of each cluster. The cluster score of each cluster will be adjusted by applying local update and global update. The system will submit a job to the most appropriate resource according to the scores.

2. VM SCHEDULING

The scheduling objectives are to improve the system's schedule ability for real-time tasks. A rolling-horizon scheduling architecture and a task-oriented energy consumption model was analyzed and built. A novel energy-aware scheduling algorithm named EARH for real-time tasks, in which a rolling-horizon policy was used to enhance the system's schedule ability. Additionally, the resource scaling up and resource scaling down strategies were developed and integrated into EARH, which can flexibly adjust the active hosts' scale so as to meet the tasks' real-time requirements and save energy.

The existing methodology presents a workload characterization of nodes by dividing tasks into task classes using the K-means algorithm. However, different from existing methods whose main objective is to understand workload characteristics, the existing system finds accurate workload characterization, while supporting task classification (e.g., labeling) at runtime. Note that machines are naturally characterized (i.e., there are 10 types of machines in the cluster). Thus, the existing solution will mainly focus on task characterization. Once the workload characterization has been obtained, the existing system introduces a monitoring mechanism that allows Harmony to capture the runtime workload composition in terms of arrival rate for each task class. To make provisioning decisions, it defines a container as a logical reservation of resources that is meant to host tasks belonging to the same task class.

1. Each task is considered as separate unit and role back.
2. To not apply vertical scaling of VM sin terms of CPU in existing energy-aware model.
3. A single task is given to a selected single cluster only.
4. Storage capacity of cluster resources is not taken into account.
5. Replication strategy is not included.

3. EASJS ALGORITHM

When science and technology advance, the problems encountered become more complicated and need more computing power. In contrast to the traditional notion of using supercomputers, Cloud computing is proposed. Distributed computing supports resource sharing. Parallel computing supports computing power. Cloud computing aims to harness the power of both distributed computing and parallel computing. The goal of Cloud computing is to aggregate idle resources on the Internet such as Central Processing Unit (CPU) cycles and storage spaces to facilitate utilization. When human culture advances, current problems in science and engineering become more complicated and need more computing power to tackle and analyze. A supercomputer is not the only choice for solving complex problems any more as a result of the speed-up of personal computers and networks. Cloud technology, which connects a number of personal computer clusters with high speed networks, can achieve the same computing power as a supercomputer does, also with a lower cost.

Enhanced Adaptive Scoring Job Scheduling (EASJS) aims to decrease job's completion time. It considers not only the computing power of each resource in the Cloud but also the transmission power of each cluster in a Cloud

system. The transmission power of each cluster is defined as the average bandwidth between different clusters. It should use the status of each resource in the Cloud as parameters to initialize the cluster score of all clusters.

Along with existing system implementation, Enhanced Adaptive Scoring Job Scheduling algorithm (EASJS) for Job splited into 'N' tasks along with Replication Strategy. In addition, jobs are divided into sub tasks and given to one or more clusters. Storage capacity requirement is also included in Cluster Score calculation.

1. Each job is designed as sub tasks.
2. A single job is given to a selected multiple clusters since jobs are split into tasks.
3. Cluster score values are recalculated even during the job is partially completed. This is achieved when a particular sub task is finished.
4. Storage capacity of cluster resources is taken into account.
5. Replication method assists in faster job completion.

In this section, the cluster score is calculated based on the following formula.

$CS_i = P \,.\, ATP_i + Q \,.\, ACP_i$

where CS_i is the cluster score for cluster i, a and b are the weight value of ATP_i and ACP_i respectively. ATP_i means the average available bandwidth the cluster i can supply to the job and is defined as:

$$ATP_i = \frac{\sum_{j=1}^{m} Bandwidth\_available_{i,j}}{m-1}, \quad i \neq j$$

where $Bandwidth\_available_{i,j}$ is the available bandwidth between cluster i and cluster j, m is the number of clusters in the entire Cloud system.

Similarly, ACP_i means the average available CPU power cluster i can supply to the job and is defined as:

$$ACP_i = \frac{\sum_{k=1}^{n} CPU\_Speed_k * (1 - load_k)}{n}$$

where CPU_speed_k is the CPU speed of resource k in cluster i, $load_k$ is the current load of the resource k in cluster i, n is the number of resources in cluster i. Also let

$$CP_k = CPU\_Speed_k * (1 - load_k)$$

CP_k indicates the available computing power of resource k.

Because the transmission power and the computing power of a resource will actually affect the performance of job execution, these two factors are used for job scheduling. Since the bandwidth between resources in the same cluster is usually very large, we only consider the bandwidth between different clusters.

Local update and global update are used to adjust the score. After a job is submitted to a resource, the status of the resource will change and local update will be applied to adjust the cluster score of the cluster containing the resource. What local update does is to get the available CPU percentage from Information Server and recalculate

the ACP, ATP and CS of the cluster. After a job is completed by a resource, global update will get information of all resources in the entire Cloud system and recalculate the ACP, ATP and CS of all clusters.

4. PERFORMANCE EVALUATION

The following Table 5.1 describes experimental result for number of social file search process in existing and proposed hit rate analysis. The table 5.1 contains number of node, existing and proposed probability VM distribution rate details are shown.

S.No.	Number of File VM Node	Existing System	Proposing System
1	50	0.266	0.321
2	100	0.279	0.287
3	150	0.315	0.328
4	200	0.348	0.352
5	250	0.389	0.397
6	300	0.395	0.398
7	350	0.413	0.420
8	400	0.434	0.449

Table 4.1 Probability VM Distribution Rate Analysis

The following Table 5.2 describes experimental result for number of video file search process in existing and proposed average delay of video file analysis. The table contains number of search social content file sharing, existing and proposed average social content file share details are shown. The following Fig 5.2 describes experimental result for number of social content file search process in existing and proposed average delay of file analysis. The figure contains number of VM Share and existing and proposed average social VM share details are shown.

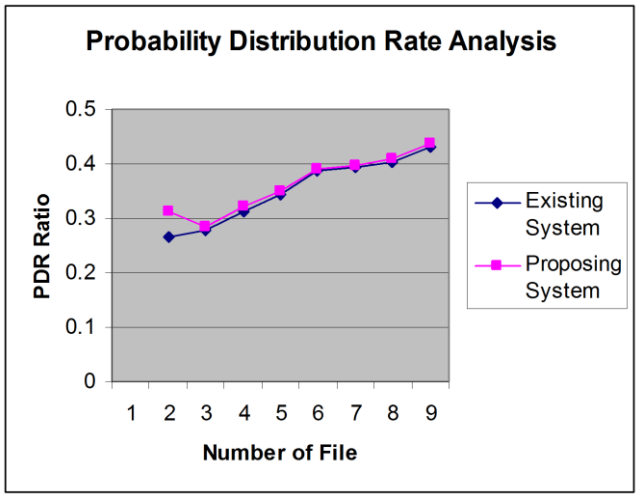

Fig 4.1 Average Probability Rate Analysis

S.NO.	Number of VM Share	Existing System	Proposing System
1	50	60.36	61.19
2	100	63.55	64.29
3	150	70.28	73.16
4	200	74.39	76.39
5	250	78.65	80.13
6	300	82.78	83.69
7	350	86.40	88.74
8	400	92.17	94.12

Table 4.2 Average VM Share Analysis

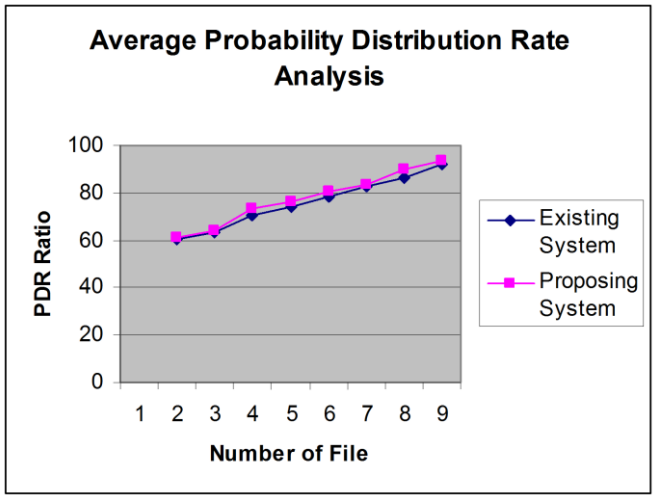

Fig 4.2 Average VM Share Analysis

5. CONCLUSION

This project proposes an adaptive scoring method to schedule jobs in grid environment. ASJS selects the fittest resource to execute a job according to the status of resources. Local and global update rules are applied to get the newest status of each resource. Local update rule updates the status of the resource and cluster which are selected to execute the job after authorize the job and the Job Scheduler uses the newest information to assign the next job. Global update rule updates the status of each resource and cluster after a job is completed by a resource. It supplies the Job Scheduler the newest information of all resources and clusters such that the Job Scheduler can select the fittest resource for the next job. The experimental results show that ASJS is capable of decreasing completion time of jobs and the performance of ASJS is better than other methods. In future, EASJS can be applied to real grid applications. This project focuses on efficient job scheduling. The project can be modified to consider division of file in data-intensive jobs. Jobs are independent, but they may have some precedence relations in real-life situation. Studying and improving EASJS for such kinds of jobs may be carried out in the future.

REFERENCES

[1] Fernandez-Baca .D, Allocating modules to processors in a distributed system, IEEE Transaction Software Engineering, volume – 34 (1989), Proceeding Page 123 -.127.

[2] Haines .S, Pro Java EE 5 Performance Management and Scalability; retrieved July 07, 2010; from IEEE Performance Management and Scalability, volume 67, (2002) , Proceeding Page184 - 189.

[3] Li .H and Buyya .S, Model-Driven Simulation of Grid Scheduling Strategies, Third IEEE International Conference on e-Science a Grid Computing ,volume 156 (2004) , Proceeding Page138 – 143.

[4] Maheswaran .M,Ali .S,Siegel .H .J,Hensgen .J,Freund .R, Dynamic matching and scheduling of a class of independent tasks onto heterogeneous computing system, Journal of Parallel and Distributed Computing 59 (1999) Proceeding Page107–131.

[5] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, Matei Zaharia, A view of cloud computing, Communications of the ACM 53 (4) (2010) Proceeding Page 50–58.

[6] Saha .D, Menasce .D, Porto .S, Static and dynamic processor scheduling disciplines in heterogeneous parallel architectures, Journal of Parallel and Distributed Computing 28 (1) (1995), Proceeding Page 1–18.

[7] Shah .S .N. .M, Mahmood .A .B and Oxley .A, Development and Performance Analysis of Grid Scheduling Algorithms, Communications in Computer and Information Science, Springer, vol. 55, Proceeding Page 170–181, 2009.

[8] Sheng-De Wang, I-Tar Hsu, Zheng-Yi Huang, Dynamic scheduling methods for computational grid environment, International Conference on Parallel and Distributed Systems 1 (2005) Proceeding Page 22–28.

[9] Syed Nasir Mehmood Shah, Ahmad Kamil Bin Mahmood, Alan Oxley, Dynamic multilevel hybrid scheduling algorithms for grid computing, Procedia Computer Science 4 (2011) Proceeding Page 402–411.

[10] William Bin Mahmood .T, Alan Oxley .H and Gunho Lee.L Grid Scheduling Dictionary Project retrieved December 05 2010; from Parallel and Distributed Computing , Vol 28 (1997), Proceeding Page 1–18.

[11] Waldspurger .W and Weihl .W .W, "An object-oriented framework for modular Resource management," in Proc. 5th Int. Workshop Object-Orientation Operation. System, 1996, Proceeding Page 138–143.

[12] Zhang .H, He .H,Chen .G, and Sun.J, "Multiple virtual machines resource scheduling for cloud computing," Application. Math. Inf. Sci. volume 7, no. 5, Proceeding page 2089-2096, 2013.

[13] Zhao.W, Wang .W, and Luo.Y, "Dynamic memory balancing fo virtual machines," ACM SIGOPS Oper. Syst. Rev., vol. 43, no. 3 pp. 37–47, 2009.