Advanced Science and Technology Letters
Vol.63 (EEC 2014), pp.6-11
http://dx.doi.org/10.14257/astl.2014.63.02

WARS: A workload-aware CPU resources scheduling for the cloud computing environment

Chao shen[1], Xiaodong Liu[2], Weiqin Tong[1]

[1]School of Computer Engineering and Science, Shanghai University, Shanghai, 200444 China
[2]School of Computer Science and Engineering, Henan Institute of Engineering, Zhengzhou, 451191 China
shenchao1010@aliyun.com,liuxiaodongxht@qq.com,qwtong@shu.edu.cn

Abstract. Virtualization-based cloud computing platforms allow multiple virtual machines (VMs) running on the same physical machine. Efficient allocation of limited underlying resources has been a key issue.This paper presents a workload-aware CPU resources scheduling method (WARS). WARS uses the allocated credits and consumed credits to diagnose the CPU resources requirements of VMs and dynamically adjusts CPU resources according to the requirements of VMs. The adjustment of CPU resources is converted into increased or decreased weights of VMs.

Keywords: Virtualization; Virtual Machine; Resource allocation.

1 Introduction

With the advantages of functional isolation, manageability and live migration, virtualization technology has been widely used in cloud computing platforms. The virtualization technology, such as xen [1], allows multiple virtual machines running on the same physical machine safely. These VMs may run different types of applications, such as processor-intensive, I/O-intensive and latency-intensive. These different applications have different resources requirements. Therefore the use of limited underlying resources has been a key issue.

However, the current virtualization technology uses static resources allocation mechanism. The underlying physical resources, such as CPU, cannot be fully utilized. Firstly, the workloads of the VM are usually varying. In order to satisfy the resources requirements of the VM, the resources of the VM must be allocated according to its peak requirement. The resources of the VM will be wasted except in peak condition. Secondly, the workloads of some virtual machine are light but they occupy physical machine resources. The idle resources cannot be used by the VMs whose workloads are heavy. Thirdly, some applications have been completed but the user may not destroy the virtual machine, so that the resources of the VM will be wasted.

This paper presents a workload-aware CPU resources scheduling method (WARS). The WARS uses the allocated credits and consumed credits to diagnose the CPU resources requirements of VMs. If the VM has not consumed

ISSN: 2287-1233 ASTL

its allocated credits in a schedule period, it means the VM needs less CPU resources. Or else, if the VM has consumed all its credits before the schedule period, it means the VM needs more CPU resources. The ratio of the consumed credits and consumed allocated credits can diagnose the actual CPU resources requirements more accurately. Based on the CPU resources requirements of VMs, the WARS will adjust CPU resources of VMs in the next schedule period. The CPU resources adjustment of WARS is converted into increased or decreased weight of VMs.

The remainder of this paper is organized as follows: Section 2 introduces related work. Section 3 describes the scheduling method. Section 4 shows the experimental results of the WARS. Finally, Section 5 summarizes our conclusions.

2 Related work

Many dynamic allocation strategies have been proposed to improve CPU resources utilization. Yuting Zhang et al. [2] proposed a self-adaptation resources allocation strategy. They used the time intervals between two consecutive virtual clock cycles (VCT) to predict resources usage. The virtual machine monitor (VMM) allocates system resources based on feedback about resources usage. Zhiyuan Shao et al. [3] used the effective average VCPU utilization rate and parallel level to diagnose resources requirements of VMs. The VMM adjusts the computing resources dynamically based on the resources requirements of VMs. It can schedule the VCPUs to appropriate physical cores by using the knowledge of the shared L2 cache architecture of the multi-core systems. Hyunku Jeong and Sung-Min Le [4] used the number of executed instructions to calculate abstract workload characteristics and dynamically allocate multi-core resources according to workload characteristics.

Fernando Rodrguez-Haro et al. [5] used Xen primitives to construct an application-aware management component that adjusts the CPU resources according to demand. By integrating the lightweight metering and controller components, they hide the low-level primitives and enhance the CPU management through high QoS parameters such as needed CPU cycles and application metrics. Jia Rao et al. Ying Song et al. [6] proposed an adaptive and dynamic resources flowing scheme amongst VMs in a VM-based utility computing environment (ADVM). VMM gets redundant resources of VMs back so that the resources can be assigned to VMs which request more resources. Evangelia Kalyvianaki et al. [7] integrated the Kalman filter into feedback controllers to dynamically allocate CPU resources

For improving the predictive ability, [8, 9] used fuzzy modeling to learn and predict the CPU resources requirements of the VM. Lixi Wang et al. [8] used adaptive fuzzy modeling to predict a database VMs multi-type resources need and allocate resources to VMs running a database serving on-demand. DynaQoS [9] is a two-layer QoS provisioning framework that supports adaptive multi-objective resources allocation. The first layer is a group of self-tuning fuzzy

controllers (STFC) that control individual objectives. When there are multiple control objectives, the second layer aggregates the requests from individual STFCs and forms a unified one.

3 WARS scheduling method

We assume there are N VMs running on the same physical machine. VM_i is the i-th VM and VC_{ij} is the j-th VCPU of the VM_i. Let w_i be the weight of the VM_i and v_i be the number of VCPUs of the VM_i. The total weight can be calculated as follows:

$$W_{total} = \sum_{i=1}^{N}(w_i \times v_i) \tag{1}$$

In order to diagnose the actual resource requirements of a VM, the WARS gathers the consumed credits and allocated credits of every VCPUs and VMs every time interval.the CPU utilization rate of the VM_i can be calculated as follows:

$$U_i(t_i) = \frac{\sum_{j=1}^{v_i}(C_{ij}^{con}(t_i))}{\sum_{j=1}^{v_i}(C_{ij}^{alc}(t_i))} \tag{2}$$

where $C_{ij}^{con}(t_i)$ represents the consumed credits of VC_{ij} at time interval t_i and $C_{ij}^{alc}(t_i))$ represents the allocated credits of VC_{ij} at time interval t_i.

If the consumed credits of the VM_i are less than its allocated credits, the CPU resources of the VM_i are abundant. Or else, the CPU resources of the VM_i are shortage. Therefore, the $U_i(t_i)$ can be used to diagnose the CPU resources requirement of the VM_i.Based on the CPU resources requirements of VMs, WARS will adjust the CPU resources of VMs.

In our WARS, the CPU resources adjustment among VMs is converted into increased or decreased weights of VMs. The CPU resources adjustment is based on the value of the $U_i(t_i)$. If the $U_i(t_i)$ is less than a threshold U_{min}, in the next schedule period, the VMM will get redundant weight of VMs back so that the weight can be assigned to VMs which request more weight. The VMM don not get all redundant resources of VMs. The VMM must guarantee VMs can work when their resources are got back, so we define the status U_{normal} which is the CPU utilization that can guarantee VMs to work. The redundant weight of the VM_i is the corresponding weight of ratio (U_{normal} to $U_i(t_i) - U_i(t_i)$). The redundant weight of the VM_i that can be calculated as follows:

$$w_i(t_{i+1}) = \frac{\sum_{j=1}^{v_i} \times v_i \times (U_{normal} - U_i(t_i))}{U_i(t_i)} \tag{3}$$

On the contrary, if the $U_i(t_i)$ is more than a threshold U_{max}, in the next scheduling period, the VM_i will request more weight to VMM. The request weight strategy is to reduce the CPU utilization $U_i(t_i)$ to U_{normal}. The requested

weight of the VM_i is the corresponding weight of ratio $(U_i(t_i) - U_{normal})$ to $U_i(t_i)$. The requested weight can be calculated as follows.

$$w_i(t_{i+1}) = \frac{\sum_{j=1}^{v_i} \times v_i \times U_i(t_i) - U_{normal}}{U_i(t_i)} \tag{4}$$

if Xen system have abundant CPU resources or VMs which need more CPU resources can be met by VMs which can provide CPU resources, the weight of VMs will be adjusted by the above formula (3)(4). Or else, there are VMs which apply to more weights but there is no idle CPU resources can be lent. WARS will reallocate CPU resources.In order to reallocate resources satisfied the above two properties, we borrow the idea of stock shares [10]. VMs which have more shares will be allocated more resources. The shares can be weight of VMs or fees paid by users. In this paper, we use the number of VCPUs to define shares of VMs. The total shares can be calculated as follows.

$$V_{total} = \sum_{i=1} N^{v_i} \tag{5}$$

The weight of VM_i which should be assigned in the next scheduling can be calculated using following formula (6-7).

$$w_i(t_{i+1}) = W_{total} \times (\alpha \times \frac{v_i}{V_{total}} + \beta \times \frac{w_i^b(t_{i+1})}{w_{total}^b(t_{i+1})}) \tag{6}$$

$$\alpha + \beta = 1 \tag{7}$$

Where $w_i^b(t_{i+1})$ represents the weight which VM_i wants to lend in the i+1 scheduling period, $w_{total}^b(t_{i+1})$ represents the total weight which VMs want to lend in the i+1 scheduling period, α and β reflect the importance of shares and resources utilization rate. The weight is assigned according to the shares of CPU resources when $\alpha = 1$. The weight is assigned according to the utilization of VMs.

4 Performance Evaluation

We have implemented WARS mechanism in xen 4.1.2 hypervisor. Our system is installed on the physical machine equipped with two Inter(R) Xeon(R) 4-core CPU running at 2.40GHz, 32G of RAM. The application running in the VMs is matrix multiply implemented by BSPCloud [10]. This section presents evaluation results for different types of application and various workloads.

In order to measure the effectiveness of WARS, we compare the schedule of WARS (WARS_xen) with the xen scheduling (Orig_Xen).

we run four VMs concurrently and all VMs are CPU resources shortage. The application running in the VM is shown in table 1. We set parameter $\alpha = 0, \alpha = 0.2, \alpha = 0.4, \alpha = 0.6, \alpha = 0.8$, and $\alpha = 1$, respectively. Fig.1 shows the experiment results. When the parameter α is big and β is small, the VMM will

allocate more CPU resources to VMs which have more weight. On the contrary, the VMM will allocate more CPU resources to VMs which have more resources requirement. α and β reflect the significance of shares and resources utilization rate. When we use parameters $\alpha = 0$, the computing time of the application run in the VM1 and VM4 declines about 15%, and the computing time of the application run in the VM2 and VM3 increases 24% and 11% respectively. With the increase of α, the computing of the application run on the VM1 increases and the computing time of the application run on the VM3 and VM4 declines. With the increase of α, the weight will become more significance in resources allocation. The weight of VM1 is smallest. So its allocated resources will less with the increase of α. Similarly, the VM3 and VM4 will be allocated more resources. We also notice that the computing time of the application run on the VM2 first declines but then increases when the parameter $\alpha = 0.8$. This is because the influence of resources utilization is more than weight when the $\alpha > 0.8$.

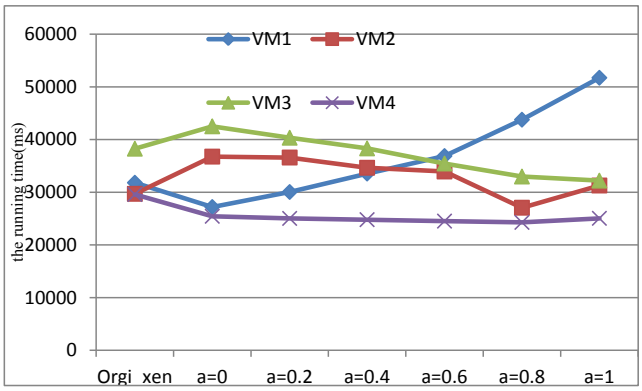

Fig. 1. xen I/O architecture.

Table 1. Application types by different VMs

VM	VCPUs	Application type	size	Used threads
VM1	2	Matrix multiplication	15541554	2
VM2	3	Matrix multiplication	15541554	2
VM3	4	Matrix multiplication	19721972	3
VM4	4	Matrix multiplication	21002100	4

5 Conclusions

In this paper, we have proposed a workload-aware CPU resources scheduling method (WARS) to improve the CPU overall utilization. WARS is a periodical allocating algorithm. With each periodical round consisting of CPU resources diagnose, request more or less CPU resources to VMM and weight adjustment. WARS uses the allocated and consumed credits to diagnose CPU resources requirements of VMs, which can improve predictive ability further. Based on the predict information, WARS will adjust the CPU resources dynamically.

References

1. P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, Xen and the art of virtualization, ACM SIGOPS Operating Systems Review, 37:164-177(2003).
2. Y. Zhang, A. Bestavros, M. Guirguis, I. Matta, and R. West, Friendly virtual machines: leveraging a feedback-control model for application adaptation, in Proceedings of the 1st ACM/USENIX international conference on Virtual execution environments, 2005:2-12.
3. Z. Y. Shao, H. Jin, Y. Li, and J. Huang, XenMVM: Exploring Potential Performance of Virtualized Multi-core Systems,Information-an International Interdisciplinary Journal, vol.14:2315-2326(2011).
4. Hyunku Jeong and Sung-Min Lee, Dynamic CPU Resource Allocation for Multi-core CE Devices Running Multiple Operating Systems, 2012 IEEE International Conference on Consumer Electronics, 2012.
5. F. Rodriguez-Haro, F. Freitag, and L. Navarro, Enhancing virtual environments with QoS aware resource management, Annales Des Telecommunications-Annals of Telecommunications, vol.64:289-303(2009).
6. Y. Song, Y. Z. Sun, H. Wang, and X. Song, An adaptive resource flowing scheme amongst VMs in a VM-based utility computing,7th IEEE International Conference on Computer and Information Technology, 2007.
7. E. Kalyvianaki, T. Charalambous, S. Hand, Acm, and Ieee, Self-Adaptive and Self-Configured CPU Resource Provisioning for Virtualized Servers Using Kalman Filters,6th ACM/IEEE International Conference on Autonomic Computing and Communications, 2009.
8. W. Lixi, X. Jing, Z. Ming, T. Yicheng, and J. A. B. Fortes, Fuzzy Modeling Based Resource Management for Virtualized Database Systems,Proceedings of the 2011 IEEE 19th International Symposium on Modelling, Analysis & Simulation of Computer and Telecommunication Systems, 2011,pp.32-42.
9. R. Jia, W. Yudi, G. Jiayu, and X. Cheng-Zhong, DynaQoS: Model-free Self-tuning Fuzzy Control of Virtualized Resources for QoS Provisioning, 2011 IEEE Nineteenth IEEE International Workshop on Quality of Service,2011.
10. C. A. Waldspurger, Memory resource management in VMware ESX Server, Proceedings of the Fifth Symposium on Operating Systems Design and Implementation, pp. 181-194, 2002.
11. Xiaodong Liu, Weiqin Tong, Fu ZhiRen, Liao WenZhao. BSPCloud: A Hybrid Distributed-memory and Shared-memory Programming Model. International Journal of Grid and Distributed Computing, 6(1) PP. 87-97, 2013

