Mechatronics and Automation Technology
J. Xu (Ed.)

doi:10.3233/ATDE221173

Study on Dynamic Resource Scheduling Method for Domestic Operating System Virtualization Based on Computing Domain

ChaoYANG[1], Pengyu LIU, Wangxin LIU, Junwei SUN and Dongyang LI
State Grid LiaoNing Information & Telecommunication (I&T) Co, Ltd, Shen Yang, China

Abstract. Virtualization of computing resources has become a mainstream technology in the computer world. This paper takes the performance optimization of resource virtualization platform on domestic operating system as the research object, deeply analyses the hardware support technology of virtualization, and the corresponding software principle and mechanism. From the point of performance research and improvement of resource virtualization, the Computer Domain (CD) based on virtualization technology is proposed. Based on the software driving principle and operating mechanism of the Virtual Manage Platform (VMP) kernel module for computational area analysis, three optimization directions are proposed: resource virtualization scheduling, I/O optimization and VM immersion optimization. The computing domain based on resource virtualization unifies the consistency of resource scheduling and improves the efficiency of resource scheduling. The experimental results show that the dynamic resource scheduling method based on computing domain can achieve significant performance improvement.

Keywords. Operating System, resource virtualization, performance optimization, computer domain.

1. Introduction

With the rapid development of large-scale integrated circuit technology, computing power will no longer become the bottleneck of business applications, business exclusive physical resources become a waste. How to utilize the excess computing resources while providing secure, stable, fast deployment and on-demand computing resource services is the mainstream topic in the current computer technology field. Virtualization technology makes these characteristics come true, and resource virtualization performance is the key indicator for users to select services [1].

Resource virtualization systems running on domestic operating systems usually include user-level tools, core virtualization, generally including the following functions: (1) memory management: including process creation and exit, pause, termination and

[1] Corresponding Author, ChaoYANG, State Grid LiaoNing Information & Telecommunication (I&T) Co, Ltd, Shen Yang, China; E-mail: yangchaoneu@sina.com

other operations; (2) System clock management: setting and acquiring time, etc. (3) Memory management: allocation and release of large pages; (4) Synchronization is mutually exclusive: providing mutually exclusive operations on shared data; (5) Interrupt and exception handling. (6) I/O operation.

Based on the above resource virtualization, this paper presents a Computing Domain (CD) technology based on resource virtualization. Computing domain refers to the abstraction of virtualized resources into an independent and complete computing domain structure [2, 3]. Each computing domain is composed of computing resources, storage resources and network resources. Based on the computing domain, resources can be easily dispatched arbitrarily throughout the computer system.

2. Computing DOMAIN Based on Domestic OS Resource Virtualization

The entire computing domain is composed of several virtual resource platforms, which are usually composed of a single computer. The computing domain can usually be made up of several virtual resource platforms (VMP). The VMP includes the following functions: (1) User-tier tools: Users can use the user-tier tools to set parameters to select different modes of resource virtualization to meet their needs. (2) Core virtualization layer: The virtualization layer is responsible for the actual creation of each virtual domain, and different modes of resource virtualization require different virtualization schemes. (3) CPU virtualization: For resource virtualization in time-sharing multiplexing mode, use processes or threads to virtualize the CPU to form a VCPU. VM Entry and VM Exit may occur within a VCPU thread. These two operations require saving and restoring resource virtualization and the critical state of the host. (4) Memory virtualization: Memory allocation is mapped linearly: Machine Address (MA) = Resource Virtualization Physical Address (PA) + Base Address Offset for each resource virtualization. (5) I/O virtualization: I/O virtualization can be driven by semi-virtualization. (6) Interrupt and exception handling: within resource virtualization, an interrupt is generated, and VM Exit is performed first. After the interrupt signal is preprocessed by the host, it is determined whether the interrupt should be handed over to resource virtualization. If resource virtualization needs to be handed over, VMP injects a virtual interrupt into resource virtualization. After VM entry, if resource virtualization finds that VMP injects a virtual interrupt, jump directly to the interrupt processing entry within resource virtualization, and wait until the interrupt is processed before doing other work.

3. Performance Optimization Technology for Virtual Resource Platforms

The nature of virtualization determines that processors must switch contexts more often than physical machines when running resource virtualization, so the performance of resource virtualization will theoretically not reach the level of physical machines. However, when processors run fast enough, users will be unaware of these performance overhead, and multiple resource virtualization increases the density of business processing [4]. It can significantly improve the overall usage of the processor. Intel VT technology improves the execution of instructions in virtualization, from software translation simulation to direct execution on the CPU, that is, the switch from VMP module running VCPU to resource virtualization is completed by hardware, which greatly improves the performance of resource virtualization.

From the above analysis of the running mode of resource virtualization, can be see that the core of virtualization is "trapped in re-simulation" of the processor. If this mode changed, it will not be able to virtualize. Therefore, the processor must switch between environments. There are three main mode switching, including resource virtualization from non-root mode to root mode switching of VMP kernel module, switching of VMP module to user space program, and switching of user empty agency to simulate device I/O processing execution system call to kernel space. Each switch involves context preservation and recovery, which incurs an overhead on execution performance. The first is a switch between kernel and resource virtualization, which is accomplished with hardware assistance and is less expensive and has little impact on performance. The other two are handled by the program, which incurs more overhead when the number of handoffs is too frequent. Therefore, in addition to environment optimization, VMP resource virtualization performance optimization should also propose optimization methods and ideas from the context switching of VCPU. Combining the previous research and analysis, this paper puts forward the following performance optimization ideas.

3.1. VMP Processing Optimization

Among the three switching modes of CPU, the switching between resource virtualization and VMP kernel module is supported by hardware. The switching between resource virtualization and VMP kernel module is implemented by hardware, which is fast and has little impact on resource virtualization performance. The switch from VMP module to user space is done by software program, which has a high system overhead. Therefore, based on analyzing the types of resource virtualization exceptions trapped, modify the handling of VM-Exit exceptions to optimize resource virtualization performance by controlling the frequency of switching between the two modes. On the one hand, dropping resource virtualization into exceptions is handled directly in the VMP kernel module to avoid the overhead of user-space program processing switching to user space. On the other hand, exceptions that do not occur frequently can be handled by user-space programs to reduce the processing tasks of the VMP kernel module and the system overhead of VMP resource virtualization.

3.2. Virtualization Process Scheduling Optimization

After resource virtualization, multiple virtual machines can run concurrently on a single physical machine. According to the rule of software operation in virtual machine, the software abnormal events are counted, the switching information of virtualization of each resource is recorded and passed to the user space monitor. The switching data is analyzed by the program or platform administrator, and the resource virtualization is compute-intensive, input/output-intensive or mixed. Then the corresponding performance optimization is implemented according to the specific resource virtualization. The job information of the resource virtualization process can be obtained when VMP starts. Based on the job information, the corresponding CPU dispatch policy can be obtained. By optimizing the dispatch policy, the resource virtualization program of VMP has higher priority than the general process in competing for CPU resources, thus improving the performance of resource virtualization. Avoid combining SI and

CGS units, such as current in amperes and magnetic field in oersteds. This often leads to confusion because equations do not balance dimensionally. If you must use mixed units, clearly state the units for each quantity that you use in an equation.

3.3. I/O Optimization

For the third switch mode of CPU, mainly resource virtualization exits to the VMP kernel module because of I/O operation requirements. Because the kernel VMP module does not handle the exceptions caused by I/O access requests substantially, it exits to the user space process processing, that is, user space performs the switch from system call to kernel space during I/O processing. If the I/O data exchange between user space and kernel does not use single processing mode, but uses I/O buffer queue to complete multiple I/O processes in one switch, the I/O processing efficiency can be significantly improved. Based on this idea, semi-virtualization drivers can be used in VMP resource virtualization to achieve the performance optimization goals of resource virtualization. Semi-virtualized drivers are implemented in primary I/O device disks and network card access drivers, thus effectively reducing I/O data replication. Resource virtualization runs in shared host memory or shares the same block of memory. Simple address remapping can avoid I/O data replication. In addition, frequent switching between root and non-root modes can be avoided. Root and non-root mode switching saves the complete data of virtual resources, reduces the switching between root and non-root modes, and effectively improves the efficiency of resource virtualization.

4. Dynamic Resource Scheduling Method For Virtualization Based On Computing Domain

Based on the virtualization of the computing domain, this paper uses the Weighted Least-Load scheduling method with weights [5, 6]. Each VPM has a specified weight in the weighted least-load scheduling method. VPMs with higher weights account for a higher percentage of active loads at any time [7, 8], and computing domain requests are allocated through the percentage of active loads on the VPM and setting weights. The workflow of the least-load scheduling method with weights is described below:

Suppose there are N VPMs, each of which has a set weight: $W_i\ (i=1,\cdots,n)$, and the value of active loads on it is $A_i\ (i=1,\cdots,n)$.

If all sums in the system $A_i\ (i=1,\cdots,n)$ are C, then C is:

$$C=\sum_{i=1}^{n}A_i \tag{1}$$

The next new load in the system is assigned to server j, which j satisfies:

$$\frac{A_j/W_j}{C}=\min\left\{\frac{A_i/W_i}{C}\right\},\ i=1\ldots n \tag{2}$$

C is a constant every time a load is balanced, so the formula (2) can be simplified to:

$$\frac{A_j}{W_j} = \min\left\{\frac{A_i}{W_i}\right\},\ i = 1 \ldots n \tag{3}$$

The least number of loads scheduling method with weights performs more division operations in the calculation than the least number of loads scheduling method, thus incurring some additional overhead. However, its load balancing effect is the best because it reflects the performance differences among VPMs by setting different weights for different VPMs.

In computing domain, VPM requests that require operating system to transmit large amounts of data are mainly read and write [9, 10]. Therefore, in order to determine the VPM's load, computing domain will record the time t and the amount D of data each VPM completes its request. Set up in time T, VPMs numbered I perform m services in total, service time is $t_i,\ i \in (1 \cdots m)$, the amount of data transferred is $d_i,\ i \in (1 \cdots m)$ in turn, then VPM numbered I load in T is:

$$L_I = \sum_{i=1}^{m} d_i \Big/ \sum_{i=1}^{m} t_i \tag{4}$$

If there are a total of N VPM, when computing domain chooses VPM, it can choose VPM numbered J to satisfy the J:

$$L_J = \min\{L_I\}, I = 1 \ldots N \tag{5}$$

The Minimum Load Method can more accurately distribute the newly generated files to the VPMs with the lowest load to achieve better load balancing. However, the minimum load method must calculate the connection time and data transmission amount of each VPM, which will impose a large additional burden on the system.

5. Conclusion

Based on the virtualization of domestic operating system resources, this paper presents a computing domain technology based on virtualized resources. Based on the computing domain technology, it can significantly provide the normalization problem of dynamic scheduling of virtual resources in domestic operating systems, and reduce the granularity of resource scheduling. This makes the Weighted Least-Load scheduling method used in this paper more accurate and efficient.

Based on the computing domain technology in this paper, VMP processing optimization, virtualization process scheduling optimization and I/O optimization are used to optimize the VMP of the domestic operating system, and their performance is compared with that of the unoptimized case as shown in table 1:

Table 1. Comparing optimization results based on computing domain optimization method.

Optimization Method	Opt-VMP	VMP	Result
VM exception handling optimization	95%	60%	With VCPU processing optimization of virtualization process, the CPU switching frequency is reduced and the system performance is improved.
I/O optimization	48.1M/s	28.3M/s	I/O efficiency has been significantly improved with semi-virtualization drivers
Scheduling optimization	9.1s	10.2s	Performance improvements after optimizing the scheduling of virtualized user processes

According to the three main ways of VMP resource virtualization performance optimization, this paper compares the performance improvement of resource virtualization in three aspects: CPU process scheduling, memory usage and I/O processing by test data before and after optimization, and verifies the feasibility of each optimization design under three performance optimization ideas. On the basis of scheduling optimization, there is still much room for improvement in the optimization results. Especially in the case of multi-resource virtualization.

Acknowledgment

This paper is supported by the scientific and technological project of the State Grid Company: Research on the Adapting Technology of the Special Operating System for Power Network based on Domestic Server Hardware(Project Number:2021YF-75).

References

[1] Anderson T, Peterson L, Shenker S, et al. Overcoming the Internet impasse through virtualization [J]. Computer, 2005, 3 (4): 34-41.
[2] Guo B J, Wang Y, Wen L Y, Lu J. Intelligent learning model based on computational area network [J]. Journal of Sichuan University: Natural Science Edition, 2013, (4): 765-769.
[3] Guo B J, Huang J, Lu J, Tang Y. Universal Computing Framework Model for Internet of Things Based on Computing Area Network [J]. Journal of Sichuan University (Natural Science Edition), 2012, 49 (01): 80-84.
[4] Tari Z, Broberg J, Zomaya A Y, et al. A least flow-time first load sharing approach for distributed server farm [J]. Journal of Parallel and Distributed Computing, 2005, 65 (7): 832-842.
[5] Elzoghdy Said S F. An Intelligent AntNet-Based Algorithm for Load Balancing in Grid Computing [J]. IJCT, 2013, (11): 2975-2986.
[6] Ruhana K. An Improved Min-min Algorithm for Job Scheduling using Ant Colony Optimization [J]. IJCSMC, 2014, 3 (5): 552-556.
[7] Bernstein P A. Middleware: A Model for Distributed System Services [J]. Communications of the ACM, 1996, 39 (2), February.
[8] Baker M A, Fox G C, Yau H W. Cluster Computing Review, Northeast Parallel Architectures Center, Syracuse University, 1995, 16 November.
[9] Cheung A L, Reeves A P. High performance computing on a cluster of workstations [C]. Proc. of the 1st Int. Symposium on High Performance Distributed Computing, 1992:152-160, September.
[10] Fox A, Gribble S D, Chawathe Y, Brewer E A, Gauthier P. Cluster-based scalable network services [C]. In Proceedings of the Sixteenth ACM Symposium on Operating System Principles, San Malo, France, 1997, Oct.