A Dynamic Memory Allocation Approach Based on Balloon Technology on Virtualization Platform

Xi Li, Ruina Chen*, Baojiang Zhang and Changtai Li
School of Information Science and Engineering, Central South University, Changsha, China, 410083

*E-mail: crn007@126.com

Abstract. In virtualization platforms, Virtual Machines(VMs) have different memory demand when running different applications. Applications with high memory demand often result in guest swapping and performance degradation of the VM. Because of the semantic gap between the Virtual Machine Monitor(VMM) and the guest Operating System(OS), it is impossible to achieve reasonable allocation and use of memory resources between the guests. Our goal is to monitor the memory information in real time and achieve memory recovery and replenish of each VM in case of memory fluctuation. This method builds up a reasonable swapping mechanism, which can achieve automatic balance of memory allocation among multiple VMs, optimize guest swapping and improve memory utilization. We display it by visualization module. Based on ballooning technology, this paper designs a dynamic memory allocation method in KVM virtualization platform, which consists of memory monitoring, memory dynamic allocation, memory resource visualization and other modules. The results of many tests and analysis demonstrate that our method has the ability to basically achieve our expected goals and requirements.

1. Introduction

The rapid development of Internet technology and computer technology has greatly promoted the evolution of new distributed computing models. From cluster computing to grid computing to the most popular cloud computing [1], their common goal is to continuously promote efficient allocation of resources. In recent years, virtualization technology has become the most critical supporting technology in cloud computing, which plays an increasingly prominent role in server integration, hardware resource allocation and other aspects. Reasonable allocation of memory resources plays an important role in the whole hardware resources, so flexible allocation and scheduling of VM resources is one of the key issues to improve the efficiency of the cloud platform [2].

The cloud platform emphasizes the flexible allocation of resources, unified management and scheduling of various resources from a variety of resource pools on demand allocation. However, multiple guest VMs are often deployed on a single physical host in virtualization platform. All the VMs have been allocated fixed memory at the beginning. But the memory requirements of each VM are different. There is often a phenomenon that one VM has free memory and the other VMs are short of memory. Furthermore, even if VMM have a large number of free memory pages, they cannot always redistribute memory to the guest in urgent need of memory to use, let alone reclaim the guest's memory pages at any time. In fact, VMM and VM are in a double-blind state because the guest OS is not aware of the presence of VMM and other VMs on the same host, and cannot request additional...
memory from VMM during run time. Similarly, VMM doesn’t have a ability to obtain memory utilization information from the guest, which is called the semantic gap [3]. The semantic gap between VMM and the guest OS leads to the realization of flexible memory in virtualization platforms [4]. In addition, the workload of multiple applications on a VM varies over time, and memory requirements may suddenly decrease or increase dramatically. When the memory requirement is less than the amount of free memory, the guest will inevitably trigger the guest paging mechanism [5], which will lead to performance degradation, memory utilization imbalance caused by waste of resources.

In this paper, we propose a dynamic memory allocation method based on ballooning technology in order to solve the above problems. This paper mainly studies the principle and implementation of memory ballooning technology in the virtualization platform of multiple guest VMs on a single physical node, and designs a dynamic memory allocation method based ballooning technology to achieve reasonable allocation of memory. This method mainly includes the monitoring module, which monitors the memory information of all VMs to obtain their memory utilization; the allocation method module, which is a dynamic memory adjustment strategy based on memory ballooning technology; and the visualization module, which is a visualization tool based on Zabbix operation and maintenance system [6]. This method has a ability to monitor every guest memory information in real time. When the guest memory pressure increases or decreases continuously, it can be supplemented or recycled to achieve the goal of automatically balancing the use of memory between multiple VMs, optimizing guest pages and improving the performance of the entire platform.

The rest of this paper is organized as follows: Section 2 briefly discusses the related research work about virtualization technology. Section 3 describes the architecture, system design and dynamic memory allocation method in detail. Section 4 introduces the evaluation and analysis on the KVM platform, and presents results using visualization tools. Finally, section 5 concludes our work and states our plans for future work.

2. Related Works
A series of researches have been carried out by industry and academia facing the problem of flexible memory sharing and allocation in virtualized cloud platform. And a variety of methods and technologies have been proposed to promote virtualized flexible memory allocation.

At present, most researches are focused on the semantic gap. Some of these research focuses on modifying the guest OS and providing a new mechanism for virtualized memory management in order to eliminate the barriers caused by the semantic gap. For example, the Active Memory Sharing technology is provided by IBM PowerVM based on IBM Power Systems and the Transcendent Memory (tmem) technology [7] [8] is first proposed by Oracle. [9] provides an API for VMM on the guest OS to enable the VMM to master the memory usage information of the guest OS, and enable the guest OS to use the services provided by the VMM. [10] provides APIs for applications that bypass the guest OS and directly request and release memory from the VMM. Although these technologies can effectively promote the sharing and flexible allocation of virtualized memory resources, their implementation and application depend on the support of specific hardware and OS versions, which seriously hinders the implementation and popularization of these technologies. Memory Hotplug [11] [12] technology supports the dynamic addition or deletion of physical memory during the operation of the OS, which increase the memory limit. However it still requires a specific version of the OS support. VMM page swapping, or host page swapping, is the temporary swapping of some memory pages to disk in the case of a tight host memory, which is the same as the traditional page swapping mechanism in the OS. It is also a typical time-for-space technology, but if the page swapping is repeated with the guest OS itself, it will result in double-paging and significant performance degradation. In addition, some research work does not modify the guest OS by inserting specific modules to take advantage of the host's free memory. For example, [13] uses the host free memory to build a singlet buffer and to be eliminated pages stored temporarily as a cooperative cache. When the IO operation hits the content being cached, the IO efficiency of the VM can be increased. More research is devoted to sharing the remaining memory on other hosts through high-speed network devices considering that the remaining
memory on a single host is often insufficient. For instance, [14] and [15] set up special memory servers in clusters, provide shared memory, and adopt hardware support to achieve guest sharing. The transparent memory page of the host OS is forwarded to the memory server.

In the research of memory elastic sharing and allocation in virtualized cloud platform, although some achievements have been made, there are still many problems to be further studied and optimized, such as the performance degradation caused by guest paging, resource waste caused by memory utilization imbalance and so on. The method proposed in this paper is based on the condition that the guest OS is completely transparent. We design a dynamic memory allocation recovery device for guest switching based on ballooning, which can realize global equilibrium allocation and scheduling of memory resources. The research of this topic is expected to significantly improve the paging performance of the guest, balance the comprehensive utilization of memory resources in the entire virtualization platform, and further improve the utilization of memory resources to reduce service costs.

3. Architecture and System Design

3.1. System Architecture
This paper designs a dynamic memory allocation method based on ballooning technology, which mainly includes three parts: memory monitoring, memory dynamic allocation and memory visualization. VMM obtains memory utilization at the same time intervals to determine the state of the client. When the guest memory is idle or busy, the balloon driver, running within the guest OS can help the VMM to reclaim or repay guest memory by inflating or deflating the balloon. And it cooperates with the designed memory dynamic allocation method to recover the guest memory. It is our goals to automatically balance the use of memory between multiple VMs, optimize guest swapping, and improve the performance of the entire platform. The whole memory change process can be rendered intuitively through the Zabbix. The specific structure is shown in Figure 1.

3.2. Memory Monitoring
We choose KVM as VMM, which uses Linux’s own scheduler to manage. In KVM management tools, libvirt, as a free open source management tool, application interface and daemon, plays an efficient role in the management of clients. Libvirt is mainly composed of application programming interface
library, libvirt daemon and virsh interactive command management tools. In libvirt, the configuration file of XML is used to describe the detailed configuration of the guest VM. Each VM has its own XML configuration file that contains the name of the VM, the number of CPUs, memory and storage, and other detailed configuration parameters and content. We can modify the configuration information in the XML to collect memory information of the VM periodically: the initial memory allocated by the VM, the memory used by the VM, the free memory unused by the VM, and the memory reclaimed by the host.

3.3. Dynamic memory allocation method
Host machine collects memory information from the virtual machine every 10 seconds. The memory utilization of the VM is defined as the proportion of memory used by VM to run business and VM memory reclaimed by the host by initial upper memory for VM configuration, that is

\[
\text{Memory}_{-}\text{Used} = \frac{(\text{Used} + \text{Loss})}{\text{Total}}
\]

(1)

Here \textit{Used} represents memory used by VM to run business. \textit{Loss} represents VM memory reclaimed by the host. \textit{Total} represents initial upper memory for VM configuration. At the same time, we define two thresholds: Memory Top and Memory Bottom. One is the upper limit of memory utilization as the critical point for replenishing memory, the other is the lower limit of memory utilization as the critical point for reclaiming memory. According to the memory utilization, we can divide the VM into three states: idle state, ideal state and shortage state. When the memory utilization of the VM is in different range, the VM has various state, and the balloon driver makes corresponding operation according to the state of the VM. The specific situation is shown in Table 1. The core idea of the method is to adjust the memory utilization of each VM to an ideal state by using memory ballooning technology. In addition, the boundary conditions are set depending on the memory of the host and VM. If the memory used by the host exceeds the upper bound, memory is no longer allocated to any VM; if the free memory of the VM is less than the lower bound to maintain its own performance, memory can not be reclaimed from the VM.

Table 1. VM state and balloon operation under different memory conditions

| Condition                  | VM State | Balloon Operation |
|---------------------------|----------|-------------------|
| Memory_{Used} > Memory_{Top} | Idle     | Deflating         |
| Memory_{Top} ≤ Memory_{Used} ≤ Memory_{Bottom} | Ideal    | -                 |
| Memory_{Used} < Memory_{Bottom} | Shortage | Inflating         |

When memory is in shortage state, the VM needs to add some memory. The Shell script calls the corresponding Memory_Comensation(). That is the memory complement function. When memory is in idle state, the VM needs to reclaim part of the memory, and The Shell script calls the corresponding Memory_Recovery(). That is memory recovery functions. These functions command to achieve the corresponding operation through the Libvirt-Virsh memory ballooning technology. When reclaiming and replenishing memory, we also introduce the coarse-grained and fine-grained concepts, which determines the strength of memory changes.

3.4. Memory Visualization
Zabbix based on Web interface is a very powerful visual operation and maintenance monitoring system. As an open source, enterprise-class, distributed system monitoring solution, it has become a popular monitoring system in the industry. Zabbix Server receives the data and status of the monitored devices, and is responsible for the project configuration, data storage, report generation and display, sending alarm information and so on. Zabbix Agent is responsible for collecting the monitoring data of the equipment and sending it to Zabbix Server. We can use the web interface in Zabbix to observe the change of memory directly. The specific structure diagram of Zabbix is shown in Figure 2.
4. Evaluation and Analysis
This section gives the implementation and evaluation of the dynamic memory allocation method. We mainly design two experiments in the whole process.

4.1. Experimental environment
We have implemented the experiments in Qemu/KVM virtualization platform. We choose Ubuntu 16.04LTS as the host and virtual machine OS. The memory of host computer is 4G. Under the KVM platform, four VMs are generated, which are called VM1, VM2, VM3 and VM4. Each VM initializes the initial memory of 1 CPU, 512MB, and the maximum memory limit is 1024MB.

4.2. The effect of ballooning technology
In this experiment, we choose VM1 and VM2 for testing. VM1 does not use memory ballooning technology for memory adjustment, and VM2 uses memory ballooning technology for memory adjustment. The two cases are compared. We configure three monitoring items on Zabbix: Total memory, Used memory and Swap memory. Total memory represents the actual memory of the guest, and the upper limit cannot exceed the maximum memory limit set when the guest was created. Used memory represents the memory used by the guest and can be used to detect fluctuations in the guest's memory pressure. Swap memory represents the swap memory of the guest, from which you can see whether the guest has paging operations and swap memory size. We choose memtester-4.2.2 to run on VM1 and VM2 terminals respectively, with 50 MB of memory pressure as the growth interval, and consuming 50 MB, 100 MB, 150 MB, 200 MB to 550 MB of memory respectively. Under different memory load pressures, observe whether VM1 and VM2 are paging and paging memory size. The memory changes and page changes are depicted in Figure 2 and Figure 3.

![Figure 2](image1.png)
(a) VM1 memory change

![Figure 3](image2.png)
(b) VM2 memory change

Figure 2. Memory change comparison of VM1 without ballooning and VM2 with ballooning

![Figure 3](image3.png)

Figure 3. Swap memory with different memory pressure
4.3. Memory allocation for multiple VMs

In this experiment, we choose VM3 and VM4 for testing. We configure four monitoring projects on Zabbix: Total memory, Available memory, Free memory, and Used memory. Available memory represents the available memory computed by the OS’s own rules and serves as a reference to the available memory of the guest. Free memory represents free memory that is not used by guests, often less than Available memory. We define the value of Memory_Top to be 60%; and the value of Memory_Bottom is 75%. The memory pressure is set to 350 MB at the VM3 terminal by memtester-4.2.2. At this time, VM3’s memory utilization is larger than 75%, VM4’s memory utilization is less than 60%. At this point, the balloon in VM4 expands, and part of the memory in VM3 is recovered; the balloon in VM3 shrinks and is allocated to VM4 memory. The memory changes of VM3 and VM4 are shown in Figure 4 and Figure 5.

![Figure 4. VM3 memory change process](image)

![Figure 5. VM4 memory change process](image)
4.4. Analysis of experimental results

According to Figure 2, VM1 does not have a guest paging operation when the memory pressure is 50M, 100M, 150M, 200M, 250M without the memory ballooning to adjust the memory. When the memory pressure increases to 300M and 350M, it starts the paging operation, and the paging volume increases continuously. This is because the actual amount of memory in VM1 remains unchanged, but memory pressure is increasing, and the VM can only start its own OS paging mechanism, paging out to swap space to alleviate memory pressure. On the contrary, VM2 does not have memory ballooning technology to adjust the situation, as the memory pressure slowly increases, the actual amount of memory also increases, paging memory has been zero, indicating that there is no guest paging operation during this period. This is because VM2 triggers the memory balloon technology adjustment mechanism under the memory balloon technology based memory adjustment strategy, and the balloon in the VM continuously shrinks to supplement the allocated memory until the VM is in a proper state or idle state. Figure 3 can also visualize a significant reduction in the number of client page changes when there is a memory balloon adjustment. It is proved that memory balloon technology can optimize the page changing operation of the guest and enhance the performance of the VM.

In Experiment 2, we selected one of the most representative cases to test. According to Figure 4, the VM3 system allocates VM3 memory because it consumes a large amount of memory, resulting in the decrease in Free memory and Available memory, an increase in Used memory and Total memory. However, after the first allocation of memory for VM3, the memory utilization is still larger than 75%, and a second allocation is made, so Total memory increases again. As shown in Figure 5, the balloon in VM4 expands and the memory is reclaimed, so Total memory drops, Free memory and Available memory drops, and the used memory remains unchanged. From the whole memory change process, we can find that the dynamic memory allocation method plays a good role in adjusting the memory of the VM, avoiding paging swap, so as to balance the memory in multiple VMs and improve the performance of the VM.

5. Conclusion

In this paper, we propose a dynamic memory allocation method based on ballooning technology. This method has the ability to balance the use of memory between multiple VMs automatically and optimize guest swapping. In our solution, memory change process is shown directly through the monitoring of the guest memory, adaptive and reasonable scheduling of memory.

In Section 4, the related experiments and results analysis are carried out. It is proved that the memory ballooning technology can optimize guest swapping. The effect of dynamic allocation memory system on multiple VM memory is obvious. The prototype system basically achieves the desired goal. The algorithm designed in this paper can be further optimized to reduce the performance degradation of the virtual machine.

Acknowledgments

I am very grateful to Peipei Zhang for her technical support. This work is supported by the National Natural Science Foundation of China (No.6160051183), Internet of things Intelligent Cloud Service Platform for unmanned stores (No.kc1701026) and Hunan financial and currency recognition and self service platform engineering technology research center.

References

[1] Jun Wu and Shou-Liang Sun 2017 A dynamic memory allocation approach for virtualization platforms 2017 IEEE 3rd international conference p 213–218

[2] Jibin Wang and Bohan Wang 2016 A Hybrid Main Memory Applied in Virtualization Environments 2016 First IEEE International Conference on Computer Communication and the Internet p 413
[3] Wei Fan, Zhujun Zhang, Tingting Wang, Bo Hu, Sihan Qing and Degang Sun 2016 Research on Security Algorithm of Virtual Machine Live Migration for KVM Virtualization System Information and Communications Security pp 54-70

[4] Liljana, Gavrilovska, Valentin Rakovic and Aleksandar Ichkov 2018 Virtualization Approach for Machine-Type Communications in Multi-RAT Environment Wireless Personal Communications 100 pp 67-79

[5] Anwar, Al-Yatama, Imtiaz Ahmad and Naelah Al-Dabbous 2017 Memory allocation algorithm for cloud services The Journal of Supercomputing 73 pp 5006 – 5033

[6] Anwar Alyatama, Asmaa Alsumait and Maryam Alotaibi 2018 Continuous memory allocation model for cloud services The Journal of Supercomputing pp 1-26

[7] D. Magenheimer, C. Mason and D. McCracken 2014 Transcendant Memory and Linux In Proceedings of Ottawa Linux Symposium(OLS) p 191-200

[8] Jayneel Gandhi, Mark D. Hill and Michael M. Swift 2017 Agile Paging for Efficient Memory Virtualization IEEE Computer Society pp 80-86

[9] Pranjali P. Deshmukh and S. Y. Amdani 2017 Survey of memory streaming techniques for virtual machine in cloud environment 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing pp 1198-1200

[10] T.-I. Salomie, G. Alonso, T. Roscoe, and K. Elphinstone 2013 Application level ballooning for efficient server consolidation The 8th ACM European Conference on Computer Systems p 337-350

[11] Pinter, S.S., Aridor, Y. and Shultz, S 2015 Improving machine virtualization with 'hotplug memory' 17th International Symposium on Computer Architecture and High Performance Computing, p 168-175

[12] Liu, H, Jin, H and Liao, X 2014 Hotplug or Ballooning: A Comparative Study on Dynamic Memory Management Techniques for Virtual Machines IEEE Transactions on Parallel and Distributed Systems pp(99): 1-14

[13] Anwar Alyatama 2018 Pricing and quantization of memory for cloud services with uniform request distribution 2018 21st Conference on Innovation in Clouds, Internet and Networks and Workshops p 1-7

[14] K. Lim, J. Chang, T. Mudge, P. Ranganathan, S. K. Reinhardt and T. F. Wenisch 2016 Disaggregated memory for expansion and sharing in blade servers The 36th Annual International Symposium on Computer Architecture

[15] K. Lim, Y. Turner, J. R. Santos, A. AuYoung, J. Chang, P. Ranganathan and T. F. Wenisch 2015 System-level implications of disaggregated memory 18th International Symposium on High Performance Computer Architecture (HPCA).