Research of Disk Migration Technology for Virtual Machine in Cloud Computing

Pei-Yun Xue Xue-Ying Zhang Jing Bai* Shu-Jin Jiao

College of Information Engineering, Taiyuan University of Technology

Taiyuan 030024, China

xuepeiyun@link.tyut.edu.cn, 236139168@qq.com, bj613@126.com, 1070584116@qq.com

Received 13 December 2012; Revised 10 July 2013; Accepted 20 July 2013

Abstract. In cloud computing, the live migration of virtual machines (VMs) can ensure the running of cloud services. In order to enhance the performance of the whole-system migration, this paper presents a new disk migration method combined the redirect-on-write (ROW) mechanism with pre-migration and hierarchical migration to complete the live migration of VMs. At the pre-migration stage, isolates the original data and update data, simultaneously partitioned target disk for redirecting into low-frequency and high-frequency blocks according write frequency. After pre-migration stage, migrates the low-frequency blocks in the next round, then, migrates the high-frequency blocks at the shutdown stage. In this way, the amount of synchronization redundant data can be reduced. Experiments show that the technology proposed in this paper can reduce the amount of new-data migrated more than 65 percent and cut down the average migration time for 24s. It performs better in the whole-system migration, especially in the case of low workload.

Keywords: cloud computing, virtual machine, disk migration, redirect-on-write, hierarchical migration

1 Introduction

The virtualization technology is one of the key technologies of cloud computing. It may achieve the rational planning and efficient management of cloud resources. The live migration of VMs from one physical host to another is a significant application to achieve load balancing, fault-tolerant of data and servers maintaining. It is an important sustaining technology to ensure the data security of cloud storage, secure running of cloud computing and credible supplying of cloud services in cloud computing environments [1].

Currently, the main tools for VM migration are used in the local area network or VM cluster, and achieved Migration of virtual machine by way of shared disk storage. Such as Xen Livemigration and Vmware Vmotion, they can implement live migration of VMs including the parameters of memory, CPU, I/O and network, and share storage devices by storage area network (SAN), net-work-attached storage (NAS) modes between the host machines instead of disk data migration. With the development of network technologies and application requirements, the whole-system live migration including disk data can not be achieved under the shared storage or WAN environments due to the large amount of the virtual machine disk data, disk data migration technology turns into the difficulties and focus of migration for VMs.

At present, the ways to achieve disk data migration of VMs are the following:

Firstly, the early shutdown-migration [2, 3] and fetching blocks to demands [4, 5]: This is a widely used way for disk migration of VMs early. Shutdown-migration requires a very long pause time. Even if Collective system does many improvements by the copy-on-write (COW) disk technology, pause time is yet longer. The other way, fetching blocks to demands, can achieve shorter pause time. But, there is a long-playing dependence on the source host after completing the migration, this may result in the waste of resources.

Secondly, the disk Migration Technology based on the combination of synchronous playback mode and pre-migration[6]: After triggering migration, operate the pre-migration of disk data firstly, simultaneously intercept all read and write requests occurred in the pre-migration process, then transmit to the destination host and save in sequence. After accomplishing pre-migration, recovering and running the virtual machine in the destination host, block disk I/O firstly until re-done all writes intercepted in the pre-migration process. This program has a short pause time. But, there may be a longer disk I/O blocking time after the virtual machine running in the destination host. In addition, the writes occurred in the pre-migration process may be repeatedly recorded. Therefore, there are large amounts of data redundancy.

Thirdly, the whole-system migration mechanism for virtual machines [7]: Migrate the whole-system of virtual machine in the round loop using pre-migration for the parameters of disk, memory, CPU, I/O and network etc, and mark the position information of update data by using of Block-Bitmap in each round of disk migration

process, then migrate in the next round. When the cycle times reach the maximum, or the update rate is greater than the transmitting rate of update data, stop the pre-migration and turn into shutdown stage, then operate data synchronization and subsequent migration. In the subsequent migration, this method may migrate all data blocks marked as dirty integrally. Consequently, there are much more repetitive data migrations, and it may result in a long migration time.

Fourthly, the disk migration technology based on real-time synchronization mechanism [8]: Through the Transmission Control Protocol (TCP) connection, transmit the update data in real time from source server to destination server, thus make two-node disk data consistent and achieve disk migration. Afterwards, a method with cycled synchronization and redundant detection was proposed in document [9] to improve the above theory, this method can reduce the amount of redundant data. The disk migration technology based on real-time synchronization mechanism is similar to the whole-system migration technology; its synchronization cycle is equivalent to the whole-system migration's pre-migration cycle, and its advantage is the synchronization cycle can be adjusted according to the I/O frequency. But, there are still a large number of redundant data to be synchronized.

In summary, there are different degrees of limitations existing in the shutdown time, amount of redundant data synchronization, total migration time and the dependence on the source server in the above disk migration technology for VMs, so it will cause a waste of resources and decline in the quality of service.

In this paper, we propose a new technology of disk migration for VMs based on the ROW technology combined with pre-migration and hierarchical migration to solve the problems of consuming too long time and migrating large amount of data.

2 Design of Disk Migration

2.1 Design Ideas

We isolate the new data from the original data of disk through the ROW Mechanism [10]. But the most applications' disk writes change a small amount of data, only account for 5% to 20% of the disk blocks [11]. Therefore, there must contain large numbers of consecutive zero when writes are redirected to the new disk blocks. While in the compression progress, we can achieve high-speed and high compression rate even though using the most simple and fast compression algorithm. In addition, most write operations are concentrated in a small amount of disk blocks, thence, we can partition the disk blocks according to write frequency and migrate respectively [12].

On the basis of Xen VM migration framework, we add the disk migration module including ROW Mechanis, WKdm compression algorithm, Pre-migration and hierarchical migration. After triggering migration, transmit the disk data from source server to destination server. At the same time, structure disk mapping relation between a reserved disk and the original disk, and found a mapping list on the source server. Then redirect the new write requests to reserved disk according to the mapping list in the first migration cycle. In the next cycle, compress the new data and transmit to destination server. Then decompress data package and synchronize in line with the disk mapping list.

2.2 Pre-migration

There are three stages including the PUSH, downtime copy and PULL in the pre-migration mechanism of VMs. Different migration tools use different combinations of the above three stages to achieve VM migration.

In this paper, we take the PUSH and downtime copy to migrate disk data. The process is shown in Fig. 1.

PUSH stages: Migrate the original disk data firstly. At the same time, monitor disk access and redirect write requests to an alternate disk. Then migrate the update data in subsequent cycle of migration to the destination host.

Shutdown copy stage: The VM is suspended on source host and has not been started on destination host. According to the disk address mapping table, read the update data, then compress and transmit to the destination host. IN this way, there will be some disk data to migrate in downtime stage, and some impact to downtime. However, experiments show that, in the case of I/O load is not too heavy, it takes a limited impact on the overall downtime.

2.3 ROW

ROW is one of the methods to achieve snapshots. We use the ROW technology to redirect requests of write and read to disk and isolate the original data and update data then transmit in batches.

There is only logical division between the original disk and reserved disk, the physical space is not necessarily independent. The key is establishing address mapping on physical address. When the mapping table is created, starting from the header corresponding to the original disk, this part of the physical unit is initialized for storing updated data, i.e. the physical address of the physical unit is equal to the logical address of the original disk.

PUSH stages: When the data is written to original disk, the system will find the corresponding physical unit according to the address of write requests, i.e. the logical address in the mapping table, and submit write requests.

Shutdown copy stage: When detect update data, the system will find the corresponding physical unit according to of the read requests in the mapping table. Then replace the logical address by the physical address of the reserved disk and submit read requests.

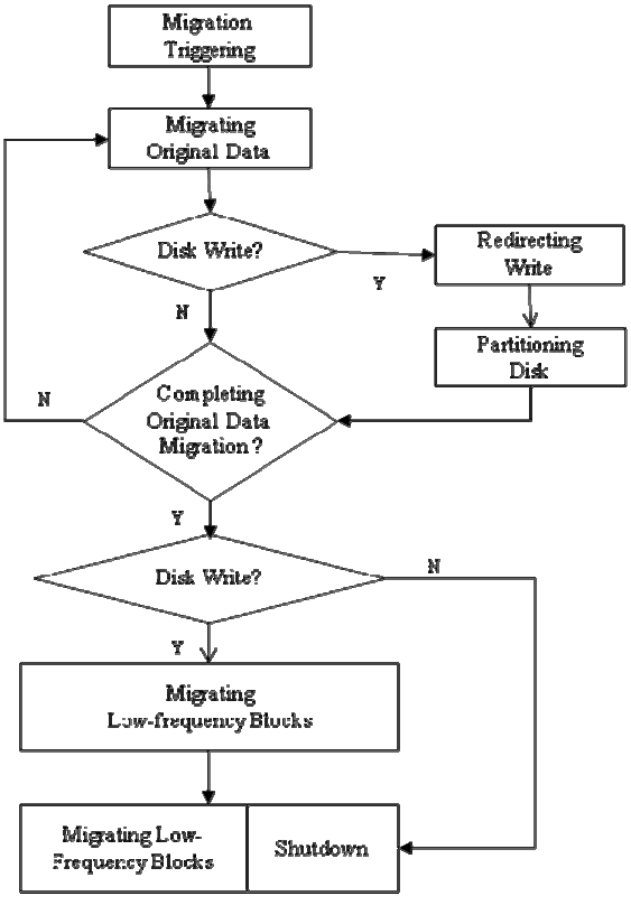

Fig. 1. Disk pre-migration flowchart

2.4 Hierarchical migration

As illustrated in Table 1, most of the workload suffer the over write more than 50% of the total write operations that concentrated in fewer disk blocks. Therefore, based on the mechanism of memory hierarchical migration, we partition reserved disk into high-frequency and low-frequency blocks in accordance with the frequency of write operation. Then migrate the low-frequency blocks after migrating the original data, and migrate the high-frequency blocks at the shutdown copy stage.

2.5 Cycle Times

In the pre-migration mechanism of VMs, will involve a problem that how many cycles we should do until the end of pre-migration, then turn into shutdown process and synchronize disk data. A classic algorithm is to compare network transmitting rate and the new data generation rate, and when the new data generation rate is the greater than network transmitting rate, reach the end of the disk pre-migration.

In this disk migration strategy, we use the ROW mechanism. After migration being triggered, redirect all of the update data to the reserved disk. If we take the above method for multi-cycle migration, the data on reserved

disk will be piled up with the increase of the cycle times, and then make the amount of synchronize data increase at the shutdown stage in the next step, and the downtime will extend.

Therefore, we adopt three loop migrations, this is, transmit all the data on the original disk to the destination server in the first round of migration, and transmit the data on the low-frequency disk in the next round. Then stop the pre-migration, transmit the data on the high-frequency disk in the shutdown stage. In this way, we can reduce the amount of data at the shutdown stage, and thus reduce downtime.

Table 1. Write Operations

Workload	Operation Type	
	Initial Write	Overwrite
MSN	31.64%	68.36%
Live Maps	99.36%	0.61%
Display Ads	12.34%	87.66%
Source Ctrl	15.93%	84.08%
Map Reduce	48.64%	51.36%

2.6 Comparison and Analysis

Fig. 2 is the comparison of the virtual machine disk migration technology based on ROW with the one based on block-bitmap in the whole-system migration.

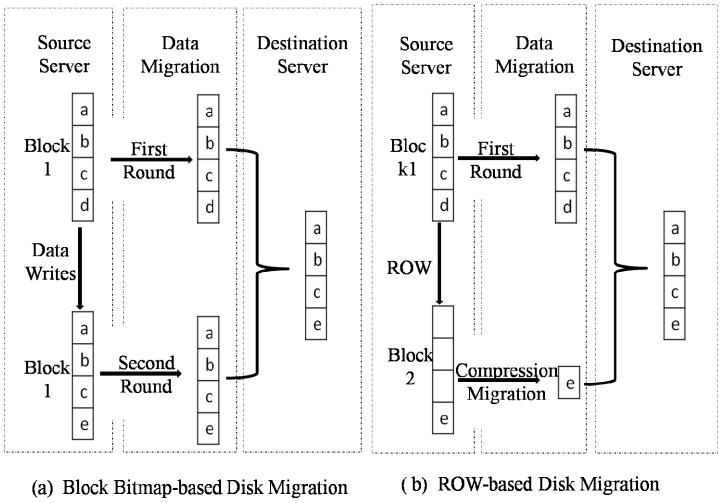

Fig. 2. Comparison of virtual machine disk migration technology

Fig. 2(a) is the block bitmap-based disk migration. Generated write requests in the first round of the migration, are written to blocks of the original disk, and then found the block bitmap to tab the disk blocks as dirty which the new data has been written on. According to the block bitmap, migrated disk blocks as a whole marked in the last round to destination server in the next round of migration.

Fig. 2(b) is the ROW-based disk migration. The data write requests generated in the first round are redirected to blocks of the reserved disk. When enter the shutdown and synchronization process, get the data through the read redirect mechanism, then compress and migrate to destination server. On the destination node, decompress the data packages and write on the corresponding disk blocks to synchronize on the basis of disk mapping list.

When virtual machine deployments are in the same experimental circumstances, we deploy the same load and cycle the same round of pre-migration so that to make both the migration generate the same amount of data in the process of pre-migration. Therefore, no matter what strategy we put to use, block bitmap-based or ROW-based disk migration, the final data on disk of destination server are “abce”.

But the migrated data are dissimilar.

In Fig. 2(a), the data are “abcd + abce”.

In Fig. 2(b), the data are “abcd + e”.

In addition, in the block bitmap-based disk migration, when we run pre-migration for more than two rounds, some disk blocks migrated in last round will be migrated repeatedly. However, the ROW-based migration can avoid this defect.

The above analysis show that, with the disk migration technology based on ROW mechanism, we can isolate the update data from the original data and transmit in batches so that to reduce the amount of migrated data thereby reduce downtime and total migration time.

3 Performance Tests

Virtual machine migration technology based on block bitmap or ROW is combined with the pre-migration mechanism to achieve the whole-system migration for VMs. We tested the above-mentioned migration strategy, enforced the whole-system migration including the parameters of disk, memory, CPU, I/O and network etc. on the same experimental circumstances, and selected the amount of migrated data and migration time to do comparative analysis.

3.1 Experiment Configuring

We set up the testing platform with three servers, one for client, the other two for source and destination servers as virtual machine host. We access the VMs through the client, and complete migration from source server to destination server. The hardware configurations are as follows: Xeon E5606 2.13GHz CPU, 4GB DDR3 memory. Allocate disk of 4GB for the virtual machines, where, 2G for original disk to save the original data, 2G for reserved disk to save the update data. The software contains Linux platform and Xen-3.0.3 virtualization software.

3.2 Experiment Design and Analysis

We conducted 5 groups experiments of the whole-system migration based on the above-mentioned two strategies, that is, 10 experiments. Set the round of disk pre-migration for three in the block bitmap-based migration. Deploy the I/O-testing program Bonnie++ on the VMs to imitate disk writes. Record the data throughput in the migration process through TCP-dump and the other performance parameters such as the total migration time and downtime.

3.2.1 Amount of Data Migration

After accomplishing migration of the original data from source server, transmit the update data generated in the pre-migration process and synchronize to destination server. The comparison of the amount of update data migrated in the two strategies is shown in Fig. 3.

The data in Fig. 3 is the average of 5 experiments. Strategy A is block bitmap-based migration. The amount of data transmitted to the destination server in the follow-up rounds of pre-migration after the completion of the first round is 176MB. Strategy B is ROW-based migration. The amount of data is 61MB. Evidently we can see that the amount of updated data in the migration based on strategy B is greatly reduced, only for 34.7% of the migration based on strategy A. The ROW-based disk migration of virtual machines can effectively avoid the synchronization of redundant data, quickly complete the disk migration in the case of limited bandwidth and improve the efficiency and availability of the VMs.

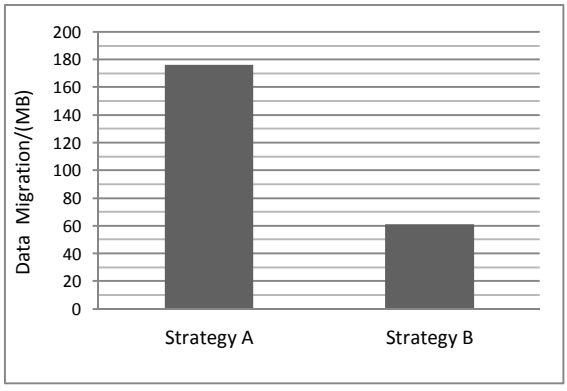

Fig. 3. Comparison of the amount of migrated data

3.2.2 Total Migration Time

The statistics of total migration time with the two strategies is shown in Fig. 4. There is obvious disparity for the total migration time. Where, the average time of the whole-system migration based on the block bitmap is 144s, and the time based on ROW is 120s. Consequently, the technology of disk migration based on ROW can shorten the total migration time.

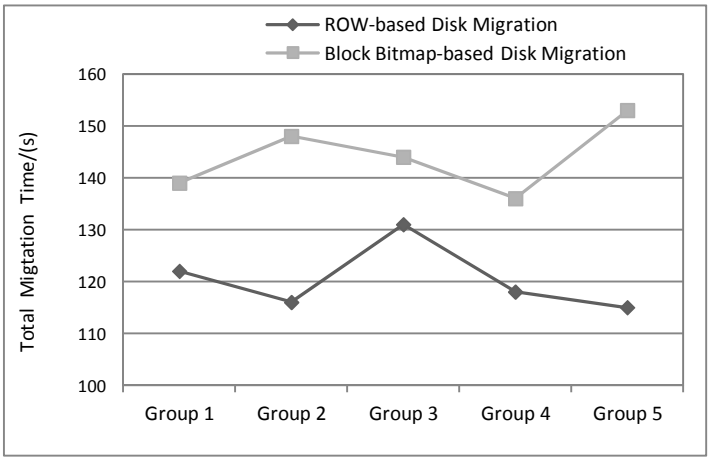

Fig. 4. Comparison of the total migration time

3.2.3 Downtime

Downtime is also one of the important indicators to measure the performance of VM migration. It directly affects the quality of service. The shorter downtime is, the harder we feel the service interruption. The statistics of downtime with the two strategies is shown in Fig. 5.

With regard to the indicator of downtime, the average time of 5 experiments with the strategy of ROW-based migration is longer, and the disparity reaches to 216ms. But this disparity is the rank of millisecond. Therefore, we can hardly feel interruption of services. The quality of services is satisfied.

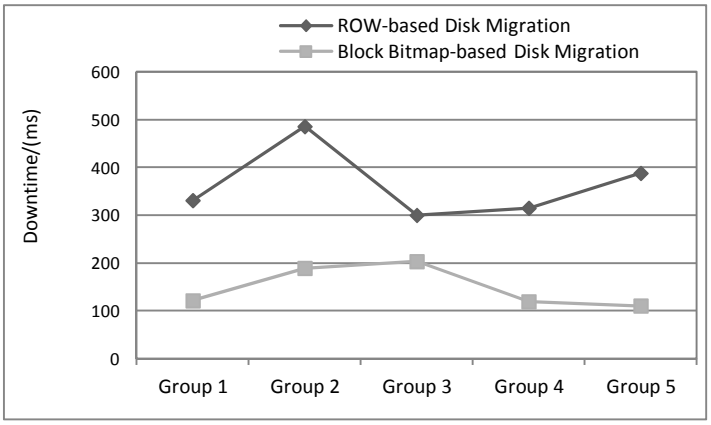

Fig. 5. Comparison of the Downtime

3.2.4 Additional Performances Analyzing

In addition, we compare and analyze the other performances of the whole-system migration based on the two strategies.

a. In the process of migration, the I/O performance of VMs drops significantly, and the descent in the migration based on ROW is more slightly obvious.

b. We changed the virtual machines' load and took the comparative experiment, then found that the higher the load on the VMs, the greater the difference in performances of migration based on the two strategies. The strategy of ROW-based migration promoted the indicators of total migration time and amount of data migration more clearly. But for downtime, the effect is more unsatisfactory. This is our next research areas for improvement.
c. There may be update data written into the low-frequency disk not to be migrated in hierarchical migration. Therefore, according to the block bitmap-based migration technology, we can add the pull phase in the pre-migration to synchronize data.

4 Conclusions

In this paper, we compared and analyzed multiple strategies of disk migration for VMs, and on this basis, designed and implemented the disk migration based on ROW. Experimental results show that the strategy of disk migration proposed in this paper can reduce the amount of migrated disk data and shorten the migration time to better meet the requirements of load-balancing and high availability in cloud computing circumstances.

Moreover, on the basis of pre-migration and ROW mechanism, we will further optimize the strategy of disk migration. Through multi-mapping lists, pre-migrate disk data alternately for several rounds, and find a more suitable opportunity of shutdown to synchronize data. In this way, achieve shorter downtime and total migration time, so that to meet the higher demand for cloud services.

Acknowledgement

This work was supported by Natural Scientific Foundation of China (No.61072087), and Scientific and Technological Research Projects (Social Development) of Shanxi Province of China (No.20120313013-6), authors gratefully acknowledge them.

References

[1] A.J. Younge, L.G. Von, L.Z. Wang, “Efficient Resource Management for Cloud Computing Environments,” in *Proceedings of 2010 International Conference on Green Computing*, IEEE Press, pp. 357-364, 2010.

[2] C. Sapuntzakis, R. Chandra, B Pfaff, “Optimizing the Migration of Virtual Computers,” in *Proceedings of 5th Symposium on Operating Systems Design and Implementation*, ACM Press, pp. 377-390, 2002.

[3] R. Chandra, N. Zeldovich, C. Sapuntzakis, “The Collective: A Cache-based System Management Architecture,” in *Proceedings of 2nd USENIX/ACM Symposium on Networked Systems Design and Implementation*, ACM Press, pp. 259-272, 2005.

[4] M. Kozuch, M. Satyanrayanan, T. Bressoud, “Seamless Mobile Computing on Fixed Infrastructure,” *IEEE Computer*, Vol. 32, No. 7, pp. 65-72, 2004.

[5] T. Hirofuchi, H. Nakada, H. Ogawa, “A Live Storage Migration Mechanism over WAN and It’s Performance Evaluation,” in *Proceedings of 3rd International Workshop on Virtualization Technologies in Distributed Computing*, ACM Press, pp. 67-74, 2009.

[6] R. Bradford, E. Kotsovinos, A. Feldmann, “Live Wide-area Migration of Virtual Machines Including Local Persistent State,” in *Proceedings of 3rd International Conference on Virtual Execution Environments*, ACM, pp. 169-179, 2007.

[7] B.B. Zhang, Y.W. Luo, X.L. Wang, “Whole-system Live Migration Mechanism for Virtual Machines,” *Acta Electronica Sinica*, Vol. 37, No. 4, pp. 894-899, 2009.

[8] T. Wood, K. Ramakrishnan, P. Shenoy, "Cloud Net: Dynamic Pooling of Cloud Resources by Live WAN Migration of Virtual Machines," in *Proceedings of 7th ACM SIGPLAN/ SIGOPS International Conference on Virtual Execution Environments*, ACM Press, pp. 121-132, 2011.

[9] Z.H. Xu, J.J. Liu, S.H. Zhao, "Disk Live Migration Strategy of Virtual Machine Based on Synchronization Mechanism," *Computing Engineering*, Vol. 38, No. 9, pp. 291-293, 2012.

[10] B. Chen, *Research on Key Technologies of on-demand Deployment of Virtual Machines in Distributed Environments*, Ph.D. Dissertation, National University of Defense Technology, Changsha, Hunan, China, 2010.

[11] Q. Yang, W.J. Xiao, J. Ren, "Trap-array: A Disk Array Architecture Providing Timely Recovery to Any Point-in-time," in *Proceedings of ACM ISCA'06*, ACM Press, pp. 289-301, 2006.

[12] S. Akoush, R. Sohan, B. Roman, "Activity Based Sector Synchronization: Efficient Transfer of Disk-State for WAN Live Migration," in *Proceedings of 19th Annual IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, IEEE Press, pp.22-31, 2011.