International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 4, April - 2013

A Survey Of Pre-Copy Live Migration Techniques In Virtual Network Environment

Neha Agrawal
M.Tech Scholar
Department of Computer Science & Engineering, MANIT Bhopal, India

R. K. Pateriya
Associate Professor
Department of Computer Science & Engineering, MANIT Bhopal, India

Abstract

Virtualization technology has vital role in cloud computing. Virtualization allows the virtual machines (VM) to create on physical hosts as well as migration of VM's data from source to destination and virtual machine (VM) seems running all the time to the client. Live migration provides the load balancing of virtual machine. One of the objectives of live migration is that it should have minimum migration time as well as downtime so that application running on VM will be suspended for negligible time. VM's data include memory, CPU data and storage disk data. In this paper we focus on various pre-copy based live migration technique in cluster environment where shared disk storage are provided, only CPU states and memory data need to be transferred.

1. Introduction

The term cloud comprises a large number of physically distributed components connected through Internet. Components include data-centre, distributed server, clients. All are connected through internet. Clients provide the end user interaction to access the cloud services. Data-center is a collection of server which contains the application to subscribe. Cloud computing can be defined as a new style of computing in which dynamically scalable and often virtualized resources are provided as a services over the internet [1]. Cloud services are broadly classified into three categories. "Infrastructure as a service (IaaS)" also known as "hardware as a service (HaaS)" provides servers, data-centres or network resources to client. Client uses the services on rent and pay only for the duration they use. Verizon's CAAS provides computing as a service, AT &T provides hosting and storage services. "Platform as a Service (PaaS)" provides hosted application environment for building cloud application. salesforce.com, Amazon EC2, Microsoft Azure provides PaaS. "Software as a service (SaaS)" provides the facility to user to access software application without installing them on user's machine. These software are managed by centralized server. In cloud computing virtualization has played an important role in resource management because it abstracts the resource such as storage and CPU through generating VM to support resource assignment [2].

A virtual machine is a software implementation of computing environment in which Operating system or program can be installed and run. VM creates its own computing environment but for the hardware resources and memory it request to underlying physical resources through virtualization layer. Hypervisor (virtual machine monitor) provide virtualization layer through which virtual machines interact with hardware as demonstrated in fig (1).

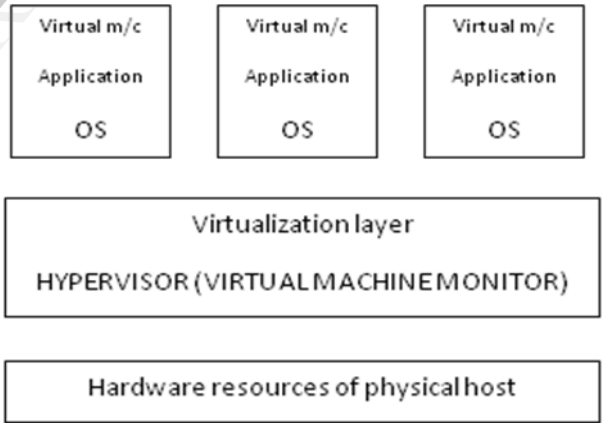


Fig.1 Virtualization of physical host

When performing VM migration two key parameters plays important role, migration time and downtime. Downtime is the time when the VM is completely stopped and doesn't provide any service to client. Migration time is the time taken by VM to perform overall migration, including downtime. For effective migration, downtime and migration time should be less. The rest of paper is organized as follows. Section II broadly classified the types of virtual machine migration. Section III describes the pre-copy based live migration techniques and

comparison between them. Section IV concludes this paper.

2. Types of virtual machine migration

There are mainly three types of migration techniques namely stop and copy based migration, post copy based migration and pre copy based migration.

2.1. Stop and copy based migration

It is a non live migration technique virtual machine is completely stopped running on source machine and all its memory pages are copied to destination machine. After copying all memory pages, VM is started on destination. Fig 2 demonstrates the stop and copy based migration.

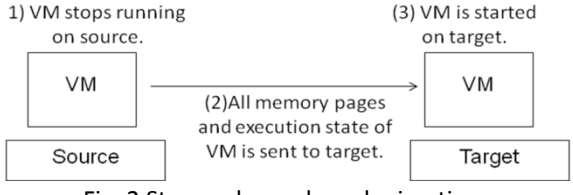


Fig. 2 Stop and copy based migration

Migration time and downtime is same for stop and copy based migration because VM is not started on target host until its all pages are sent to target. In fig [2] time taken by step (1) to step (3) is total migration time and also downtime. Drawback of this method is that VM's services are completely unavailable until it is started on destination causes increased downtime. Post-copy and pre-copy based migration technique described below is live migration techniques.

2.2. Post-copy based migration

In Post-copy VM stops running on source and only its execution state (CPU, register and non pageable memory) is sent to target machine and VM start running on target even the entire memory pages have not been transferred and still resides on source. When VM need any memory page it generates page fault and that corresponding page is sent from source to target machine. When all memory pages are transferred to target machine, VM is completely started on target host. Fig 3 demonstrates the post copy based migration.

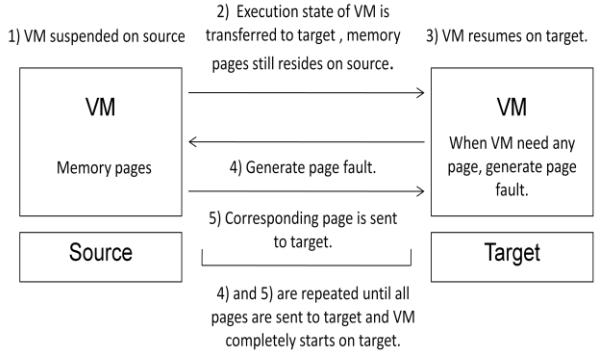


Fig. 3 Post-copy based migration

2.3. Pre-copy based migration

Pre-copy method is generally used for live migration. In the first round it transfers all the memory pages to destination machine then iteratively copies pages modified in last round. Process is repeated until the writable working set (WWS) becomes small. When WWS becomes small it performs stop and copy and transfer CPU state and dirty pages. WWS contains the pages modified in each round.

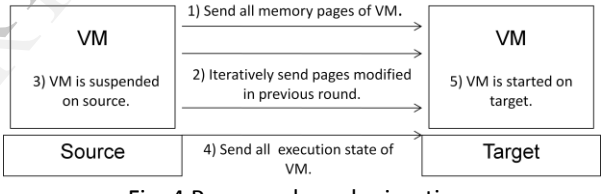


Fig. 4 Pre-copy based migration

In fig [4] time taken by step 3 to step 5 is downtime and overall time including all steps is total migration time. Standard pre-copy method doesn't provide the facility to record the frequently modified pages thus when dirty data generation rate (high frequently modified pages) is faster than memory page transfer rate, migration can be failed. Pre copy is more reliable than post copy in case of destination failure. In pre copy the source node still keeps the up to date copy of all memory data of VM and CPU state, so VM can be recovered if necessary, from source but in post copy VM cannot be recovered because destination contains the up to date copy of VM.

3. Pre-copy based migration techniques

There are mainly three types of pre-copy based migration techniques given below.

3.1. Improved pre-copy based approach

Improved pre -copy method [3] is a modification to traditional pre- copy approach described above, since it

provides the facility to keep the records of frequently modified pages. This approach uses 3 types of bitmap, To-send, To-skip & To-fix. To-send bitmap contains the pages modified in previous iteration. To-skip contains the pages modified in current iteration. To-fix contains the pages which are fixed to be sent in last iteration. One more bitmap to-send-last is used. To-send-last bitmap contains the pages which are appeared in to-skip bitmap; these pages are frequently modified pages and sent in last iteration.

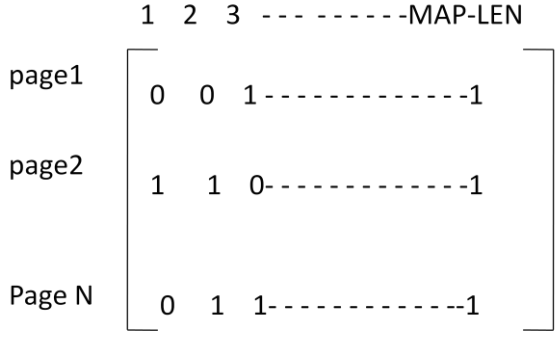
To-send	0	1	1	0
To-skip	0	1	0	1
Send or not	no	no	yes	no

This approach is useful to reduce the no of iterations by minimizing the unnecessary transfer of frequently updated pages. Problem with this approach is that it depends only on two bitmap To-send & To-skip to identify the frequently updated pages which are useless when the page is modified alternatively, like 101010 then only the last modified copy of page need to be send but it is sent 3 times.

3.2. Matrix bitmap algorithm

Improved pre- copy method (A) are not appropriate when page is modified like this 101010... , as discussed above. Matrix bitmap algorithm [5] uses many bitmaps to determine whether to send or not a corresponding page, that's why it is named as matrix bitmap algorithm. Many bitmaps are collected in iteration. A variable MAP-LEN is used which determines the number of bitmaps collected at one iteration.

Dirty bitmap collected MAP-LEN times



Here 0 determines that page is not modified, 1 determines page is modified in corresponding iteration. after collecting the bitmap, page weight is calculated at each iteration to find whether weight is greater than threshold or not. If the page weight is greater than threshold page won't be sent in this iteration.

Suppose MAP-LEN is 4, and page is modified like this 1110 for 4 times then read from least significant bit to most significant bit (0111) and weight(0111)2 =(7)10.If weight is greater than threshold then page won't be sent to destination. Threshold is determined by $2^{MAP-LEN-1}$.

Success of algorithm depends on variable MAP-LEN. MAP-LEN is used to find the threshold value which determines the high dirty pages. It should be appropriate value to get better result than previous pre-copy based approach. In any workload where modifying rate of page is low, MAP-LEN should be high value.

3.3. Time series based pre-copy approach

In previous method only two dirty bitmap are used to record the dirty page. Modifying condition of page is determined only by To-send and To-skip. Page is sent to destination only when To-send is set to 1 and To-skip is 0.Time series based approach [4] modifies the previous approach by adding a time series array of historical bitmap named as To-send-h, To-send-h array records the history of pages. According to the previous records of page, it is decided whether to send or not the page in this iteration. To-send-h stores the values of To-send bitmap in previous iterations. Pages are divided in to two categories, High dirty pages and low dirty pages. If the page is satisfying below equation is called high dirty page otherwise low dirty pages.

$$\sum_{i=1}^{N} (p \in \text{To} - \text{send} - \text{h[i]}) >= K \quad \text{.....(1)}$$

Where N is size of array 'To-send-h' denotes time series of bitmaps and K denotes threshold value. To-send-h[i] stores the value of To-send bitmap of pages in i^{th} iteration. To-send-h[i] determines whether the page p was modified or not in iteration i. In each iteration i we check the history of page p from 'To-send-h' array whether it is appeared or not in previous iterations. If the number of appearance of page p is more than threshold, it is called high dirty page and is sent in last iteration.

Time series based work as follows (For any page p)

```
IF (To-send=1 & To-skip=1 OR To-send=0 & To-skip=1
Or To-send=0 & To-skip=0) THEN

 Don't send page p to destination.

IF(To-send=1 & To-skip=0) THEN

   Check the history of page p from equation (1) to
    determine whether it is high dirty page or not.

 IF it satisfies equation (1) THEN

          Don't send the page p to destination.

ELSE
```

Success of Time series based approach depends on the value of threshold. Value of threshold should be decided according to type of workload. In heavy workload environment where Virtual machine is continuously performing write operation to memory, value of threshold should not be so high, otherwise unnecessary high modifying pages will be transferred until reach the threshold value.

3.4. Two phase strategy

Typical one phase strategy (OP) [3] sends the page when page is dirtied in previous iteration but not in current iteration i.e. to-send is set as 1 and to-skip is 0. Two phase strategy [6] uses the combination of one phase and second chance (SC) strategy first phase follow the SC strategy and 2nd phase follow the typical one phase strategy.

Second chance (SC) strategy- SC provides the second chance to the dirty page to be duplicated in destination. Dirty page will be copied to the destination only when it is kept clean for two consecutive iterations i.e. After being dirtied its bitmap is 0 for next two consecutive iterations. Two phase (TP) strategy avoids the duplication of frequently updated pages by giving them second chance in first phase. Working of two phase method as follows:-

1. Start the migration according to Second chance strategy.

2. Send the dirty page to destination only if page is kept clean for two consecutive iterations.

3. IF (28 iterations have been carried on OR no. of dirty pages<55 OR no. of duplicated pages exceeds 2 and half of size of VM)

THEN

Switch to one phase (OP) strategy.

When any one of the 3 phase -changing conditions is satisfied, SC strategy is switched to typical one phase (OP) strategy to typical one phase (OP) strategy. Optimal number of iteration for SC strategy in the 1st phase of two phase strategy lies between 1 and 29 but only 28 iterations have been taken to perform in SC strategy after that SC strategy is switched to typical OP strategy and perform stop and copy. In second condition when number of dirty pages are less than 55 then switch to typical OP strategy and perform stop and copy because the number of dirty pages is small and is a one of terminating condition used in one phase(OP) strategy. In third condition when number of dirty pages become more than 2 and half of size of VM, then switch to 2nd phase and perform iterations according to typical OP strategy otherwise it will take large amount in 1st phase and downtime will also increased. When third condition occurred, we can't get benefit of second chance strategy.

3.5. Pre copy using memory compression

Memory compression is also one of the techniques to reduce the migration time and downtime. Virtual machine's memory pages are compressed and then transferred to destination. [7] Uses the characteristic based compression (CBC) algorithm to compress the memory pages. Memory pages are compressed according to characteristic of data it contains. Property of a good compression algorithm is that it gives higher compression ratio and low overhead. Overhead is generally a time taken by CPU to compress the memory page. Compression ratio denotes the amount of compressed data. It is a ratio of freed data after being compressed to original data. But both parameter overhead and compression ratio are contradictory. It is difficult for compression algorithm to provide both low overhead and high compression ratio at a time so proper balancing is required between compression ratio and overhead and according to characteristics of memory page's data it is decided whether to use compression algorithm with high compression ratio or to use algorithm which produce low overhead. According to this algorithm memory data are classified in to 3 categories.

(1) Pages comprises of many zero bytes and very few non zero bytes. If the no. of zero bytes exceeds the threshold of zero bytes then only the offset of non -zero bytes and its value is sent to destination.

(2) If the page contains great many of similar words. When the number of similar words exceeds the threshold of similar words then compress the page using the algorithm that will take little time (low overhead).

(3) Otherwise compress the page using the algorithm with high compression ratio.

Problem with this method is that it requires overhead to compress and decompress page. There are some unused and zero pages in memory but they are also compressed and sent to destination which are unnecessary to transfer. Some identical pages are also there. There is no need to send all identical pages but

they all are sent which increases the overhead and migration time.

3.6. Memory ballooning

One of the techniques to reduce the migration time is to don't send the unused memory pages to target host. It will decrease the migration time and reduce the number of iteration. Typical ballooning used in Xen balloons the unused pages only when there is not enough memory for new VM. In Dynamic self ballooning (DSB) [8], VM continues balloons the unused memory over its execution lifetime. Memory ballooning saved a large amount of memory. But only the ballooning is not a solution. Ballooning removes only unused pages some identical pages are also there, they all are sent to destination still causes long Migration and downtime.

3.7. Trace/replay technique with CPU scheduling.

Previous methods described above uses the dirty pages to transfer. Trace/replay method [9] uses the log files of source VM instead of dirty pages. Benefit of sending log files instead of dirty pages is, reduced downtime and migration time.log files are that contains a log of events which affects the execution of system. It keeps recently running information of VM.

At the first round checkpoint is transferred to destination and then log files of Virtual machine in consecutive round. Checkpoint/recovery technique provides the system the ability to tolerate failures, failed execution is recovered to earlier safe state. During the generation of checkpoint VM stops running and a snapshot of VM having states of all program are saved on source host, after that VM is started running and checkpoint is transferred to target host. After 1st round log files are transferred iteratively in subsequent rounds. Target host performs log replay. Two threshold values of log files are used. High threshold value S_c and low threshold value S_f. When the size of log generated on source host or log accumulated on destination host exceeds the high threshold value $S_{c,\ CPU}$ scheduling is performed source host and CPU quantum allocated to that VM is reduced. When we reduce the CPU quantum for VM, log generation rate will also reduce. When the size of log files reaches below the low threshold value S_f, stop and copy is performed and the last log file is transferred and replayed on target host.

In this method it is considered that log generation rate is linearly proportional to CPU quantum allocated but some other parameters like process priority are skipped. They also affect the relationship between log generation rate and allocated CPU quantum.

4. Conclusion

Live migration is a transfer of Virtual machine data from source to target without disrupting the services running on virtual machine, as possible as.post copy and pre copy both are live migration methods.post copy takes long migration time. Pre copy is better than post copy. In this paper various pre-copy based migration techniques are discussed still some modification are needed to existing live migration methods.

5. Acknowledgement

The Success of this research work would have been uncertain without the help and guidance of a dedicated group of people in our institute MANIT Bhopal. We would like to express our true and sincere acknowledgements as the appreciation for their contributions, encouragement and support. The researchers also wish to express gratitude and warmest appreciation to people, who, in any way have contributed and inspired the researchers.

6. References

[1] Borko Furht, Armando Escalante, *Handbook of Cloud Computing*, Springer Science+Business Media, NewYork, 2010.

[2] C. Clark, K. Fraser, S. Had, J.G Hansen, E. Jul, C. Limpach, I. Pratt, A. Warfield. "Live migration of Virtual Machine," in *Proc. The 2nd conference on symposium on Networked Systems Design and Implementation*, Denmark, 2005, pp.1-14.

[3] F. Ma., F. Liu and Z. Liu. "Live Virtual machine Migration Based on Improved Pre-copy Approach," in *Proc. Software Engineering and Services sciences*, Beijing, 2010, pp.230-233.

[4] Bolin Hu , Zhou Lei, Yu Lei, Dong Xu, Jaindum Li. "A Time Series Based Approach for Live Migration of Virtual Machines," in *Proc. IEEE 17th International Conference on Parallel and Distributed System*, China,2011,pp.947-952.

[5] W. Cui and M. Song. "Memory Migration with Matrix Bitmap Algorithm," in *Proc. IEEE 2nd Symp. Web Society*, Beijing, 2010, pp.277-281.

[6] Cho-Chin Lin, Yu-Chi Huang and Zong-Dejian. "A Two Phase Iterative Pre-copy Strategy for Live Migration of Virtual Machines," in *Proc. ICCM*, Taiwan, 2012, pp-29-34.

[7] Hai Jin, Li Deng, Song Wu, Xuanhua Shi, Xiaodong Pan," Live virtual Machine Migration with Adaptive Memory Compression," in *Proc. cluster computing and workshop*,China,2009, pp.1-10.

[8] Michael R. Hines and Kartik Gopalan," Post-copy Live Migration of Virtual Machines using Adaptive Pre-paging and Dynamic Self Ballooning," in *Proc.*

ACM/Usenix international conference on virtual execution environments, SUNY, 2009, pp.51-60.

[9] W.Liu and Tao Fan et al, "the Live Migration of Virtual Machine Based on Recovering System and CPU Scheduling," in *Proc. ITAIC*, China,2011, pp.1088-1096.