ENHANCED TIME SERIES BASED PRE-COPY METHOD FOR LIVE MIGRATION OF VIRTUAL MACHINE

Neha Agrawal[1] & R.K. Pateriya[2]
[1]M.Tech Scholar, Dept of Computer Science & Engineering, MANIT, Bhopal, India
[2]Associate Professor, Dept of Computer Science & Engineering, MANIT, Bhopal, India

ABSTRACT

Virtualization technology plays a great role in cloud computing. Virtualization supports creation and migration of virtual machines (VM) on physical hosts. 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. In this paper we proposed an improved time series based live migration technique which modifies the existing time series based algorithm and provide the second chance (SC) to pages before sending to destination. This approach is very useful in reducing total number of duplicated pages in various iterations, resulting in reduced migration time and total migration time.

I. INTRODUCTION

Virtualization technology is one of the key factor in area of cloud computing. In virtualized environment multiple applications can be run at a same time. Virtualization allows the creation of virtual machine on physical host. Virtual machine monitor (hypervisor) is used to create multiple virtual machines (VM) on host. Each virtual machine is having their own operating system (O.S.) called guest operating system. Unlike the multiprogramming, in virtualization CPU is shared among the operating systems. Fig 1 shows the framework for virtualization of physical host. One of the benefits of virtualization is that live migration of VM is possible. Live migration is useful in load balancing, server consolidation or power saving. We are not focusing on applications or benefits of live migration but on the approaches to improve the performance of migration. Now the question comes how to perform migration? Two types of live migration approaches are used by hypervisor. First is post-copy based migration and second is pre-copy based migration.

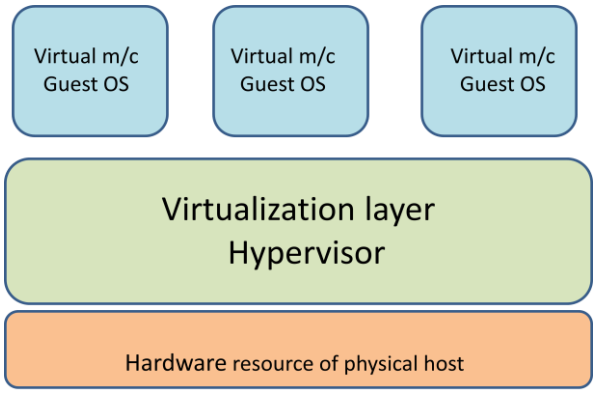

Fig. 1 Virtualization of physical host

In post-copy, as its name implies, copy the virtual machine memory pages at the later stage. First it migrate the virtual machine's execution states which are necessary to start the virtual machine on target but all memory pages still resides on source and only on the demand of virtual machine particular page is sent to target. Pre-copy technique is opposite to post-copy; first it sends all virtual machine memory pages to target whereas virtual machine is still running on source then performs

various iterations to send updated pages to target through iterations. In last round it performs stop and copy, all execution states of virtual machine is migrated and virtual machine is started on target. Section 3 describes the pre-copy technique.

Performance of live migration depends on two metrics. Migration time and Down time. Migration time is the time when virtual machine is providing services but pages are migrating to target. Down time is the time when virtual stops running and its execution states are sent to target. Total migration time is a sum of migration time and downtime. Virtual machine data include memory data; CPU data and local disk storage data. In this paper we use pre-copy based live migration technique. Our approach combines the Time Series based approach [4] and Two-phase strategy [6] for live migration but slightly modifications are done.

II. RELATED WORK

Several algorithms have come to improve the performance of virtual machine live migration. Memory ballooning [2] balloons the unused memory pages and only useful pages are migrated. Memory compression algorithm [7] uses the characteristic based compression (CBC) to compress the memory pages. According to the word similarity and byte similarity different compression algorithms are applied. Compressed pages are transferred to target but some unused and identical pages are also sent which are useless to transfer. Matrix bitmap algorithm [5] uses multiple bitmaps to identify the frequently modifying pages. A variable MAP_LEN determines how many numbers of bitmaps is used to identify high dirty pages and also threshold value. Success of algorithm depends on variable MAP_LEN. Hierarchical copy algorithm [1] creates hierarchy of memory pages and classifies them in to levels of clean pages, pages without high dirty rate, and pages with high dirty rate. Live migration of virtual machine with memory exploration and encoding [10] categorize the allocated and unallocated pages .Unallocated pages are not sent and allocated pages are encoded using run length encoding (RLE) encoding algorithm and then transferred to target. Check pointing/recovery and trace /replay technology with CPU scheduling [9] transfer log files instead of dirty pages. Log replay rate must be faster than log generation rate otherwise downtime may occur. Two phase strategy [6] as the name implies uses the two phases, first phase follow the concept of second chance (SC) strategy. In SC, if the page is clean for two consecutive rounds after being dirtied then it is sent to target, otherwise sent in last round. Second phase follow normal pre-copy strategy, if the page is clean after being dirtied it is sent to target. Time series based approach [4] use the historical bitmap; it uses a time series array to record history of page. If page is modified more number of times than the specified threshold page is declared as high dirty page and is transferred in last round. Success of algorithm depends on size of threshold value. If the appropriate value of threshold is not chosen it may not give better result than standard pre-copy method. Our proposed work take a concept from both time series based approach and two phase approach to reduce the migration time of virtual machine.

III. PRE-COPY BASED APPROACH

Pre-copy strategy for virtual machine migration is most common technique and better than post-copy strategy in case of destination failure. As described above pre-copy method first send copy of all memory pages of virtual pages to destination and virtual machine still runs on source and updated pages are iteratively send to target. When the number of updated pages reaches below the specified threshold or number of specific iterations have been completed then virtual machine is stopped on source, all its CPU state and execution state are copied to destination and virtual machine starts running on target host.

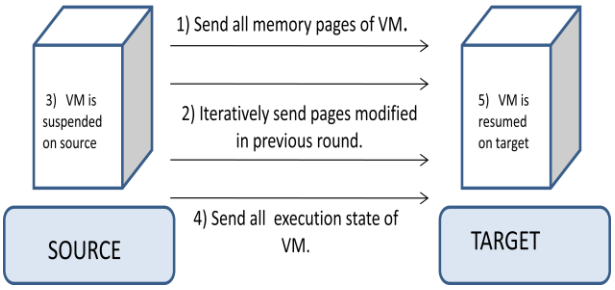

Fig. 2 Pre-Copy based migration

Standard pre-copy approach uses two bitmap to_send and to_skip to identify the high dirty pages. to_send records pages modified in last iteration ,to_skip contains pages modified in current iteration. Whenever the value of to_send =1 and to_skip=1 or to_send=0 and to_skip=1, pages are considered as high dirty pages and are sent in last round. But the standard pre-copy is dependent only on two bitmap to identify frequently modifying pages, so the probability of accurately finding high dirty pages is low. In this paper we are considering two techniques Time series based techniques and Two-phase strategy, both techniques use the concept of pre-copy.

3.1. Time series based approach

Time series based approach [4] enhances the standard pre-copy approach. Standard pre-copy approach identifies the high dirty pages on the basis of only two bitmap to_send and to_skip as described above but a time series based approach uses an array of bitmaps 'to_send_h' of size N. This array is used to record the last N history of pages. If any page is modified in any iteration i then the value of to_send_h is set as 1 for that particular page in iteration i otherwise set as 0. In every iteration time series array is checked, if any page is modified more number of times than the specified threshold K, page is declared as high dirty page for that iteration and to be sent in last round. Performance of this algorithm depends on parameter threshold K and size of time series array N, for a better result than standard pre-copy method, appropriate ratio of K/N should be chosen.

3.2. Two-phase strategy

Instead of taking historical records of pages like in 'Time series based approach', 'two phase strategy' [6] identify high dirty pages by giving them second chance (SC) to the page. In first phase value of to_send and to_skip is checked, whenever the value of to_send =1 and to_skip=0 for a particular page, page is given a second chance(SC) if it is kept clean for a second time then it is sent to target otherwise declared as high dirty page. First phase follow the second chance (SC) strategy while Second phase follow the normal pre-copy strategy. It does not give second chance (SC) to pages. The number of iterations, number of dirtied pages and number of duplicated pages are checked. When the number of dirtied pages is less than 50 or number of duplicated pages exceeds the predefined threshold of virtual machine or 28 iterations have been completed, switching is performed from one phase to second phase. Working of two-phase strategy as follows: [11]

First Phase:

1. Start the migration according to Second chance (SC) strategy.
2. Send the dirty page to destination only if page is kept clean for two consecutive iterations.

Switching condition from one phase to second phase: 28 iterations have been carried on OR no. of dirty pages<55 OR no. of duplicated pages exceeds 2 and half of size of virtual machine.

Second phase: send pages for which to_send=1 and to_skip=0.

Both the time series based method and second chance (SC) strategy use the different concept to identify the high dirty pages. One method identifies the high dirty pages on basis of the history of

pages while the other identifies on the basis of future record by giving a second chance to page. Our proposed approach combines the both method and slightly modifies the time series based approach.

IV. PROPOSED WORK

Modified Time Series Based Approach: Modified time series based approach eliminates the problem of taking threshold value k similar to time_series approach Bitmap to_send and to_skip are used. to_send contains the pages modified in previous iteration. to_skip contains the pages modified in current iteration, an array to_send_h with size N is used to record the history of page in last N iterations, if any page is modified in particular iteration then value of array to_send_h is set to 1 for a particular page in a particular iteration otherwise set to 0. Problem with time series based method was that it was dependent on threshold value K, if the appropriate ratio of K/N is not chosen, it cannot give better performance. Instead of depending on threshold value K, our approach counts the number of zeros and number of ones in array to_send_h, if the probability of occurrence of ones in array to_ send_h is more than zeros then page is high modifying page.

$$\sum_{i=1}^{N} \textbf{\textit{Count One}}(\textbf{\textit{to\_send\_h}}[i]) \geq \sum_{i=1}^{N} \textbf{\textit{CountZero}}(\textbf{\textit{to\_send\_h}}[i]$$

Equation (1)

CountOne and CountZero functions count the number of ones and zeros in array to_send_h. Pages for which value of to_send =1 and to_skip=0, we check the the history of page through array to_send_h[N]. If the page satisfies the equation (1) is declared as high dirty page and sent in last round. When the page is not satisfying equation (1), it is given a Second chance (SC).

Applying second chance (SC) to pages: After applying modified time series approach, Pages selected to be sent are passed through second chance (SC) strategy, if page is given a second chance and if it is kept clean for second time then it is sent to target. Fig.3 explains our approach.

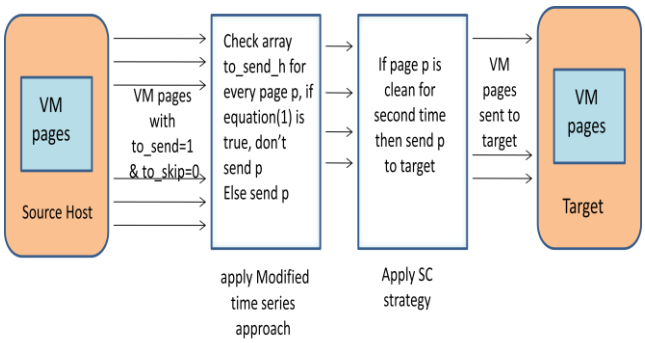

Fig. 3 Modified time series approach

We have taken an advantage of both time series and two-phase approach to reduce migration time. Following algorithm describes our approach.

Algorithm:

Input: N: size of array to_send_h
to_send_h : time series array
to_send : dirty bitmap of previous iteration
to_skip : dirty bitmap of current iteration

Send all VM's memory pages in first time.

```
to_send  ← dirty pages;
to_skip  ← NULL;
i ← 0;

 While (True) {

      Get dirty bitmap of Virtual Machine (VM);

       For each page p {
          IF  (to_send= =0 & to_skip= =0 )  OR
              (to_send= =0 & to_skip= =1)   OR
              ( to_send= =1 & to_skip= =1)  then

                    Continue;

           Else IF (to_send==1&to_skip==0) THEN
                  IF equation (1) is true THEN

                                        Continue;

                ELSE
                 Give a second chance (SC) to page p;
                        IF page is kept clean for two consecutive round THEN

                                              Send page to target host.

                        ELSE
                              Continue;
}

              to_send_h[i] ←   to_send;
              i←   (i+1)%N;
              to_send ←   to_skip;
              Update to_skip;

          IF (last iteration & to_send= =1) THEN
             send page p to destination;
             break;

}
```

Analysis of Algorithm

Our proposed algorithm combines the history and future record of pages to identify frequently modifying pages; it is obvious that our approach provides more restrictions to avoid the unnecessary transfer of dirty pages in iterations. Only the pages, satisfying equation (1) and clean for two consecutive round, are sent to target. Algorithm is suitable for both low dirty page environment and high dirty page environment and avoid taking any threshold value as an input. Effect of our algorithm on various migration parameters can be easily analyzed.

Pages transferred in iterations: only the page satisfying both condition is sent to target so reducing the pages transferred in various iterations.

Migration time: less number of pages is transferred so time taken by pre-copy iterations will also be reduced.

Down time: Down time is the time when Virtual machine stops running on source and its dirty pages and execution states are transferred and virtual machine starts running on target. Down time may be slightly increased as compare to standard pre-copy. It may depend on environment of dirty pages.

Total migration time: Total migration time is the time taken to perform iterations and downtime. Total migration time will be reduced in our approach.

V. RESULTS AND DISCUSSION

We have tested our algorithm on java platforms using netbeans. The size VM is 128 pages. A dirty matrix is used to denote the pages modified in various iterations. The row of dirty matrix denotes the number of VM pages and column denotes the iterations number. 30 random patterns of dirty matrix are generated with value of matrix either 0 or 1. If the value of matrix is 0 for a particular page in particular iteration, means the page is kept clean for that particular iteration otherwise denotes a dirtied page for that particular iteration.

Analysis of migration time and down time: We have compared the migration time of time series based approach and our approach for 30 random patterns of dirty matrix. Less number of pages is transferred in our approach so time taken by pre-copy iterations will also be reduced. Fig 4 demonstrates the comparison of migration time for our approach and time series based approach

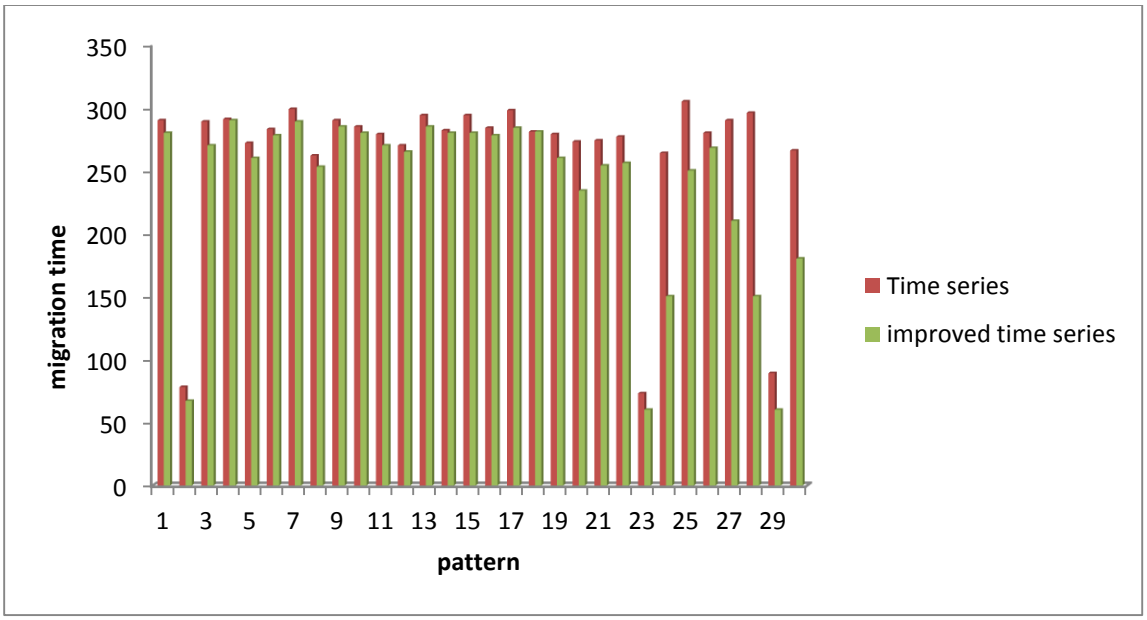

Fig. 4 analysis of migration time for time series based approach and our approach

Down time is the time when Virtual machine stops running on source and its dirty pages and execution states are transferred and virtual machine starts running on target. Down time is slightly increased as compare to standard pre-copy. It is because the pages are sent in iterations only when they are kept clean for two consecutive rounds after being dirtied so the number of pages for last round is slightly increased causes increased down time. It may depend on environment of dirty pages. Fig. 5 demonstrates the down time for time series approach and our approach.

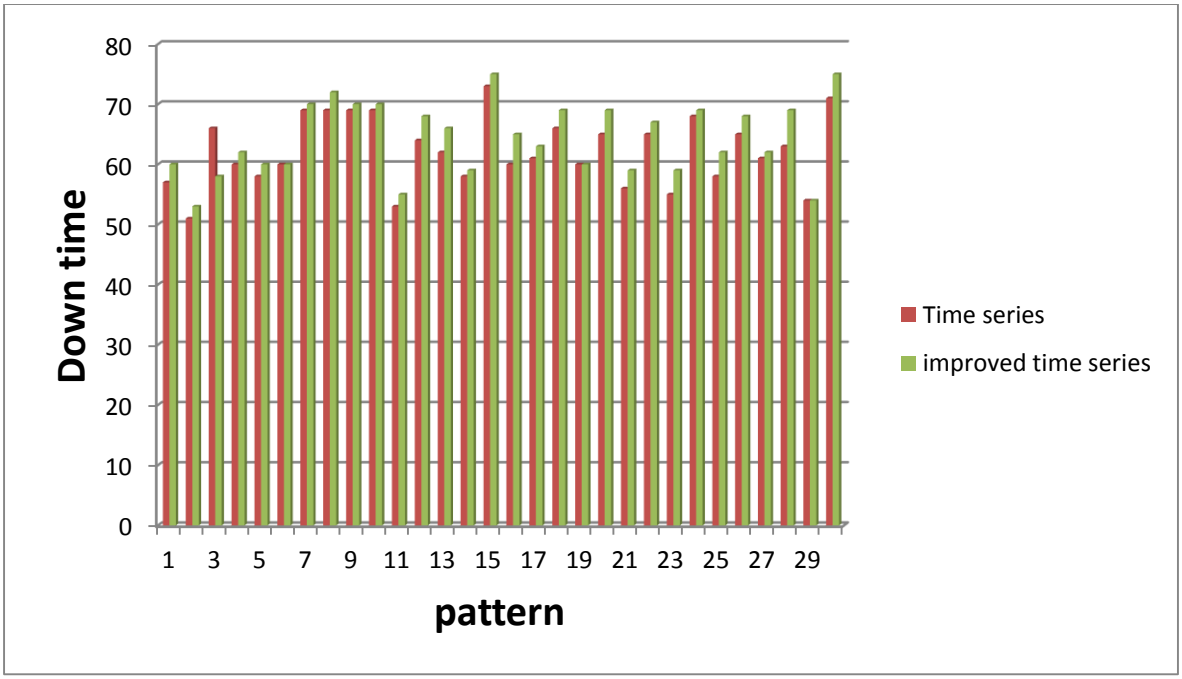

Fig. 5 analysis of down time for time series based approach and our approach

VI. CONCLUSIONS AND FUTURE WORK

In this paper we have analyzed the pre-copy based live migration techniques and combined time series based pre-copy method and two-phase strategy to get features of both techniques. Some modifications are done on existing time series based technique to enhance the performance of live migration. Definitely total number of pages transferred will be reduced; will result in decreased migration time but downtime may be slightly increased as compare to standard pre-copy technique. In the future our proposed work will be enhanced to achieve the minimum down-time as compare to existing approaches.

ACKNOWLEDGEMENTS

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.

REFERENCES

[1]. Zhaobin Liu, Wenyu Qu, Tao Yan, Haitao Li, Keqiu Li, "*Hierarchical Copy Algorithm of Xen Live Migration,"* Proc. International conference on cyber- enabled distributed computing and knowledge discovery.2010,pp.361-364.

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

[3] F. Ma., F. Liu and Z. Liu, *"Live Virtual machine Migration Based on Improved Pre-copy Approach,"* Proc Software Engineering and Services sciences, 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,"* IEEE 17th International Conference on Parallel and Distributed System, 2011,pp.947-952.

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

[6] Cho-Chin Lin, Yu-Chi Huang and Zong-Dejian, *"A Two Phase Iterative Precopy 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 cluster computing and workshop,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 of the ACM/Usenix international conference on virtual execution environments,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,"* J. Network and Computer Application, vol.34, Issue 4, 2011,pp.1088-1096.
[10] Yanqing Ma, Hongbo Wang, Jiankang Dong, Yangyang li,, Shiduan Cheng*,"ME2:Efficient live migration of virtual machine with memory exploration and encoding,"* IEEE international conference on cluster computing,2012,pp.610-613.

AUTHORS

Neha Agrawal has completed her B.E. in Computer Science from Bhilai Institute of Technology, Durg, CG, India in 2010 & presently she is pursuing M.tech in Information Security branch from Maulana Azad National Institute of Technology, Bhopal, MP, India

R.K. Pateriya is presently designated as Associate Professor in Information Technology department of Maulana Azad National Institute of Technology, Bhopal, MP, India. His qualification is Phd, M.Tech & BE.