Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 3, Issue. 2, February 2014, pg.779 – 787

RESEARCH ARTICLE

Enhanced Live Migration of Virtual Machine Using Comparison of Modified and Unmodified Pages

Sushil Kumar Soni[1], Ravi Kant Kapoor[2]

[1]Computer Technology & Applications, India

[2]Associate Professor, Computer Engineering & Application, India

[1] rec.sushil21@yahoo.com; [2] rkkapoor@nitttrbpl.ac.in

Abstract— Now a days cloud computing is one of the fast growing technology in the field of computer science and information technology because of online, cheap and pay as use scheme. The cloud computing is mainly a business-oriented model to provide on demand computing resource. It has become popular in short time because of its attractive services like easy to use, pay as use and accessibility of its services throughout the world ~~etc~~. The cloud computing concept is motivated by the idea that information processing can be public utility and can be done more efficiently on large farms of computing resources and storage systems with the availability of all time throughout the world accessible via the Internet. Virtual machine migration is one of the crucial activities that are carried out in cloud management. In this paper, we have proposed a model to remove some overhead in migration approach to increase its efficiency. The model is implemented and tested using simulator and results are compared with the contemporary approaches of migration.

Keywords— VM migration; live migration; Virtualization; Cloud; Data Canter; Data-Broker

I. INTRODUCTION

In the cloud computing, the computing resources are provided to the client through virtualization, on the internet. The large scale computing infrastructure is established by cloud providers to make availability of online computing services in flexible manner so the user find easiness to use the computing services [1]. According to NIST [3,4] cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources. The computing resources include networks, servers, storage, applications, and services. In cloud computing, the shared pool of computing resources can be rapidly provisioned and released [2]. The management effort or service provider interaction for cloud user is also minimized to increase easiness. This cloud model is basically composed of five essential characteristics, three types of service models, and four deployment models.

In this paper, we have discussed the virtual machine migration overview with proposed model to remove some overhead in migration approach. In section 2, we have discussed the background knowledge of the virtual machine migration with their types. Section 3 presents brief of literature survey of migration techniques. In section 4, we have presented the proposed model. Section 5 discuss about performance evaluation and simulation result with different parameter. Section 6 gives the concluding remarks of the study carried out.

II. VIRTUAL MACHINE MIGRATION

VM migration [5] is one of the important approaches in the area of physical machine virtualization, which allows application to be transparently migrated along with their execution environments across physical machines. Virtual machine migration is needed for load balancing, server consolidation (power saving) and resource scheduling. For effective migration, downtime and migration time should minimum.

A) Stop and copy based migration

It is a non live migration technique. Virtual machine completely stop running on source machine and all its memory pages are copied to destination machine. After copying all memory pages, VM is started on destination. Migration time and downtime is same for stop and copy based migration because VM does not start on target host until its all pages are sent to target. Drawback of this method is that VM's services are completely unavailable until it is started on destination, causes increased downtime.

B) Live migration

Live migration of virtual machine [5] allows the VM to be migrated almost without any interrupt to its application's execution. VM migration is an important means for managing applications and resources in large virtualized systems. It enables resource usage to be dynamically balanced in the entire virtualized system across physical host boundaries and it allows applications to be dynamically relocated to improve performance and reliability.

i) Post-Copy Based Migration

In Post-copy [5,6], VM stops running on source and only its execution state (CPU, register, memory pages necessary to start VM on target) is sent to target machine and VM starts 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. In this method the page fault is one of the overhead.

ii) 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 operations and transfer CPU state and dirty pages. WWS contains the pages modified in each round.

III. LITERATURE SURVEY

A) Improved Pre-Copy Approach [7]

A modification to traditional pre-copy approach has come in 2010. F. Ma, F. Lui and Z. Lui have given improved pre-copy method for live migration [9]. It 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

The improved pre-copy migration procedure as used by Xen, includes the following stages in turn: pre-migration, reservation, iterative pre-copy, stop and copy, commitment and activation. The stages stop and copy and commitment contribute to the downtime of VM migration, while total migration time includes downtime and the duration of the stage iterative pre-copy.

In improved pre-copy approach, those frequently updated pages are stored into the to_send_last bitmap page, and transmitted only in the last iteration, so the WWS can converge quickly, and the iterative process can be

finished in several iterations. This approach is useful to reduce the number of iterations by minimizing the unnecessary transfer of frequently updated pages. Fig.4.2 describes the improved pre-copy approach.

Improved pre-copy method performs well in reducing number of iterations and so reduced total data transferred. Improved pre-copy approach gives less data transferred during migration so reduced total migration time but down time is increased as compare to traditional pre-copy approach because the frequently modified pages are also sent in last round. 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.

B) Live Migration of virtual machine based on full system trace and reply [8]

This approach 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: 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 first round, log files are transferred iteratively in subsequent rounds. Target host performs log replay.

In each iteration, log generated on source host is compared with predefined threshold V_{th}. Whenever the log generated on source are reduced to V_{th}, source host asks the target host whether to perform stop and copy or not. When the target host doesn't replay fast i.e. the log accumulated on target are larger than V_{th}, source host wait and doesn't perform stop and copy. When the logs generated on source as well as log accumulated on target is reduced to V_{th}, stop and copy is performed. Success of this algorithm depends on following two conditions:

- The rate of dirty data generation should not be higher than data transfer rate, otherwise process of live migration will fail.
- Log replay rate must be faster than log generation rate otherwise downtime may be longer than the time transferring checkpoint from source to target host and the process is not convergent which results in last stop and copy round takes much time.

C) VM Migration based on recovery system and CPU scheduling [9]

An additional concept of CPU scheduling is added. Unlike the full system trace and replay, it uses two threshold values of log files. 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 on 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. Working of an algorithm is as follows.

i. **Initialization**: Source host makes a request of migration to target host. If target host is ready to receive migrated VM, it will respond with message to the request and the migration will begin.
ii. **Make checkpoint and transfer it:** An image file of source VM which has the states of system, virtual memory, CPU registers, and virtual disk, is generated on source host and transferred to target host.
iii. **Iteratively transfer log files**: In the first round, the checkpoint is copied from source to target host, while the VM source is running and events are recorded in a log file. Target host is replaying with received log file once it had recovered from the checkpoint. In consecutive iterations log file generated during the previous round is transferred. The iterative process is convergent if the log is replayed faster than it generated.
iv. **CPU Scheduling**: This is an optional phase. When one of the two conditions or both, the size of log generated on source host or accumulated on target host larger than Sc that is the threshold size of log files, CPU scheduling is performed to reduce the percentage processor quantum allocated to source VM.
v. **Stop-and-Copy**: When the size of log files on target host is less than a low threshold value or the number of iteration reaches N, the maximum number of iteration, the source VM is suspended, and then the last log file will be transferred and replayed.

vi. **Service Takeover**: After target host informs source host that it has synchronized their states, source VM can be discarded. The migrated VM running on target host is the primary host now and takes over the services offered by source VM.

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.

D) Time series based approach [10]

Time series based approach given by Bolin Hu et.al in 2011 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.

E) Two-phase strategy [11]

Instead of taking historical records of pages like in 'Time series based approach', 'two phase strategy' by Cho-chin Lin et.al. in 2011 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.

IV. PROPOSED MODEL

Proposed algorithm is based on pre copy based approach for live migration of virtual machine. It eliminates the problem live migration of virtual machine by comparing dirty bits of virtual machines taking phase wise migration to get benefits of phase wise migration.

In Pre-copy approach of virtual machine migration, only two variables are used to keep the record of dirty page. As we know that, the virtual machines modify their pages time to time. So the pages, those are continuously modified, are send to destination using stop and copy approach in the last phase. The modifying condition of page is determined only by to-send and to-skip bitmap. Page is sent to destination only when to-send is set to 1 and to-skip is 0.

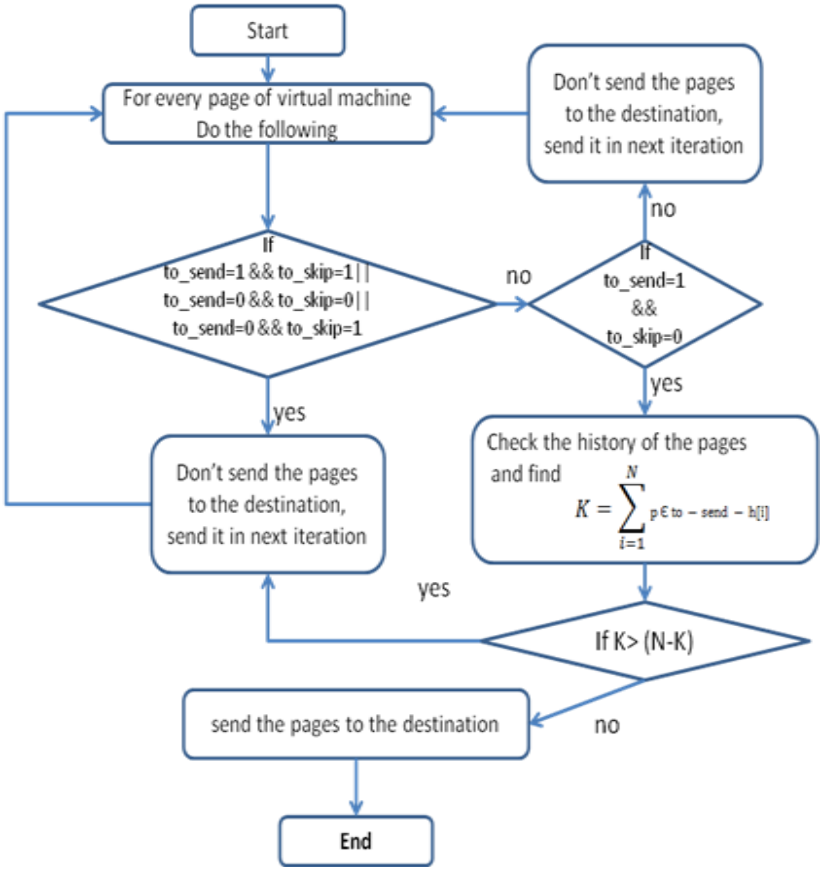

Fig. 4.1 virtual machine migration model

This approach modifies the previous approach by adding a history 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.

$$K = \sum_{i=1}^{N} p \in to - send - h[i]$$

Where N is size of array 'to-send-h' denotes historical record of bitmaps and K denotes the value of the number of modified pages in the current iteration. Variable 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 un-modified, it is called high dirty page and is sent in last iteration. Fig.5.1 gives the iterative process of time series based pre-copy method.

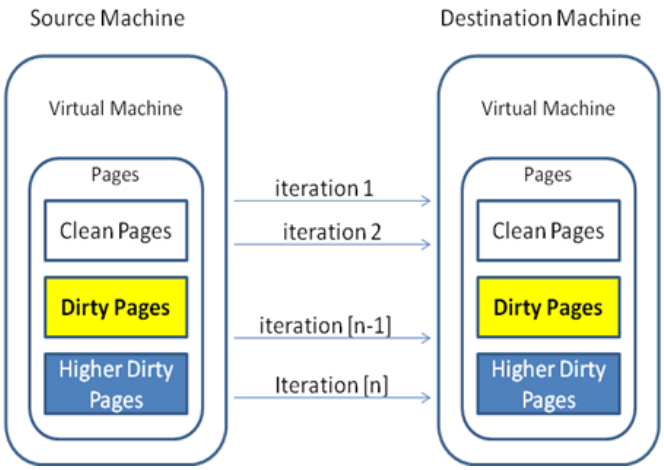

Fig. 4.2 Iterative process of vm transferring

Algorithm 1: New_VM_Migration_Approach ()

Initialize Number of Pages=m, to_skip=0, to_send=0,
num=number of iteration, to_send_h[i].

[1] for i=1 to num
[2] for each pages of VM do
[3] if (to_send=1 && to_skip=1|| to_send=0 && to_skip=0|| to_send=0 && to_skip=1)
[4] Don't send the pages to the destination, send it in next iteration.
[5] if (to_send=1 && to_skip=0)
[6] Check the history of the pages
[7] Calculate $K = \sum_{i=1}^{N} p \in \text{to\_send\_h[i]}$
[8] if K <= N - K
[9] Transfer page to the destination
[10] else
[11] Don't send the pages to the destination, send it in next iteration.
[12] End for
[13] then perform stop and copy operation to send the pages to destination
[14] end if
[15] end for

Success of the approach depends on the value of dynamically modification of virtual machine pages. In previous method of pre-copy based migration, the 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. In our proposed method we count the modified pages and unmodified pages. If the modified pages are more than un-modified pages than the action is to be taken for migrate the virtual machine.

Proposed algorithm combines the history and future record of pages to identify frequently modifying pages; it is obvious that proposed approach provides more restrictions to avoid the unnecessary transfer of dirty pages in iterations. Only the pages, satisfying equation 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 proposed algorithm on various migration parameters can easily analyze.

V. PERFORMANCE EVALUATION

Proposed pre-copy based migration algorithm for live migration of VM is implemented using CloudSim simulator, it eliminates the problem of taking dynamic behavior of cloud application. To check the effectiveness of proposed algorithm, comparisons are done between proposed approach and already developed approach with respect to different performance parameter.

Pages transferred in iterations: Only the page satisfying both condition transfer to the target so reducing the pages transferred in various iterations.

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 same or 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 reduced in our approach.

Simulation Setup and Measurement

Proposed algorithm tested on CloudSim simulator. A dirty matrix is used to denote the pages of virtual machine modified in various iterations. Only two VM are taken to implement our method. Row of dirty matrix denotes the number of VM pages and column denotes the number of history pages taken into each VM pages. 40 random patterns of dirty matrix are generated with binary value of matrix. If the value of matrix is zero 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. For each random pattern of dirty matrix, I have checked the performance of my proposed algorithm without threshold value and graph is plotted for different performance parameter (migration time, down time and total migration time).

Experiment and Simulation Result Graph

CloudSim simulator is use to implement our method. We are using a random function to generate the matrix which show the modify and unmodified pages. Since we are using a random matrix , so to plot the graph we are executing our and the previous method 5 times for the same parameters. Following environment are use to plot the graph.

For the proposed method

The total number of virtual machine = 2
Number of pages in each VM= 20
Number of history consider in each page=5
Number of iterations before stop and copy=5

For the previous method

The total number of virtual machine = 2
Number of pages in each VM= 20
Number of history consider in each page=5
Number of iteration before stop and copy=5
Threshold=5

Analysis of total number of page transmitted: To compare the total number of page transmitted, page transfer in stop and copy phase, total transmission time and down time in the migration process, we are executing both the method 5 times.

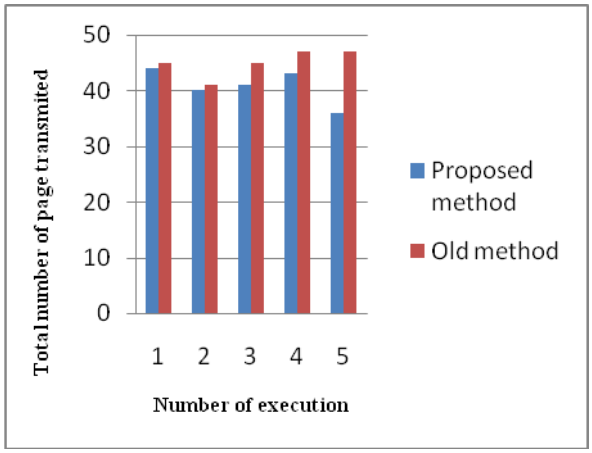

Fig 5.1 Total No. of page transmitted

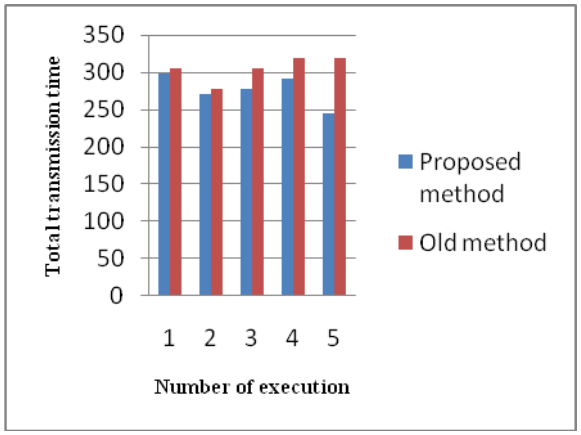

Figure 5.2: Total migration time

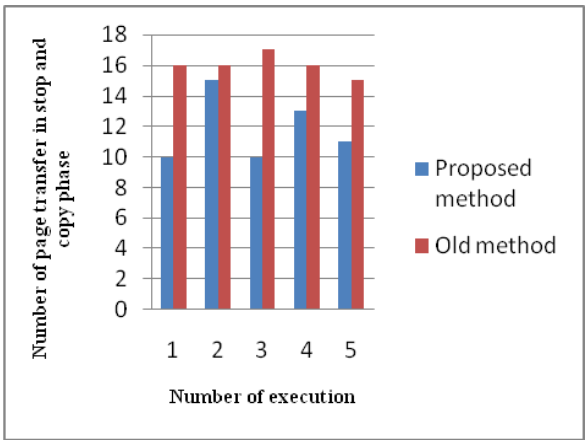

Figure 5.3: Page transfer in stop and copy phase

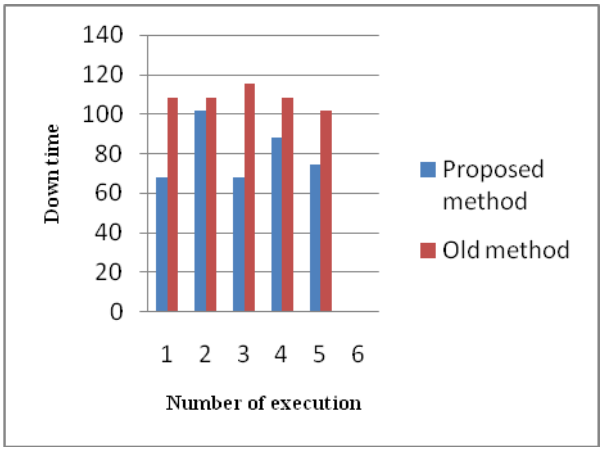

Figure 5.4: Down time

CONCLUSION

Total migration time and down time play an important role in the VM migration. This work provides an enhancement on existing pre-copy based live migration techniques and presents a new idea for live migration of virtual machine in cloud computing environment, which reduces the time taken to migration of virtual machine or application from one physical host to another host. Proposed Approach modifies the existing "Time Series Based Approach". It eliminates the problem of taking threshold value and compares the number of modified pages and unmodified pages. It improves results by reduced virtual machine downtime and total migration time.

REFERENCES

[1] Yatendra sahu, R.K. Pateriya "cloud computing overview with load balancing techniques" International Journal of Computer Applications (0975 – 8887) Volume 65– No.24, March 2013, pp: 40-44.

[2] Peter Mell,Timothy Grance, "Cloud Computing" by National Institute of Standards and Technology - Computer Security Resource Center-www.csrc.nist.gov.

[3] Wenke Ji, Jiangbo Ma "A Reference Model of Cloud Operating and Open Source Software Implementation Mapping" in 18th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises, 2009.

[4] Miyuki sato"creating next generation cloud computing based network services and the contribution of social cloud operation support system (OSS) to society" 18th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises, 2012.

[5] 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, pp.1-14, 2005.

[6] B. Prasad Rimal, Eunmi Choi, "A taxonomy and survey of cloud computing systems," *on 5th international joint conference on INC, IMS and IDC, published by IEEE Computer Society*,2009.

[7] F. Ma, F. Liu, Z. Liu," Live virtual machine migration based on improved pre-copy approach ," In Proc. Software engineering and Services science, pp. 230-233, 2010.

[8] H. Jin, H. Liu, X. Liao, L. Hu, P. Li," Live migration of virtual machine based on full system trace and replay,"

[9] W. Lio, Tao Fan et.al, " Live migration of virtual machine based on recovering system and CPU scheduling*," In Proc. ITAC, China*, pp. 1088-1096, 2011.

[10] 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, pp.947-952, 2011.

[11] 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.

