

Live Migration of Virtual Machines using Mirroring Technique

Ezhilarasie Rajapackiyam, Arvind Venkatesa Subramanian, Umamakeswari Arumugam

School of Computing, SASTRA Deemed University, Thanjavur, India

Article history

Received: 08-07-2019

Revised: 09-10-2019

Accepted: 12-03-2020

Corresponding Author:

Umamakeswari Arumugam
SASTRA Deemed University,
School of Computing,
Thanjavur, India
Email: a_umamakeswari@yahoo.com

Abstract: Virtualization is a key concept in empowering the “Infrastructure as-a-Service (IaaS)” of cloud-based services. Live migration of virtual machines is the process of moving virtual machine from one physical host to another without interrupting other VMs. Migration of VMs supports the improved efficiency of resource usage and dynamic resource provision capabilities. The performance metrics of live migration are down time and total migration time. In this paper, a technique to reduce the total migration time of standard memory migration using the mirroring of VMs at destination host is proposed. It mainly concentrates on the total migration time and maximum profit of providers. The proposed technique avoids the iterative transmission of memory pages from the source host to destination host. Performance results depict that the proposed algorithm can reduce 36% of the data transmitted during migration and 26% of the total migration time when compared with pre-copy migration technique.

Keywords: Virtualization, Virtual Machine (VM), Live Migration, Pre-Copy Technique, Total Migration Time

Introduction

Cloud computing (Josep *et al.*, 2010) has been envisioned as the current generation architecture of IT enterprise. It provides infrastructure-as-a-Service (IaaS) (Bhardwaj *et al.*, 2010) to the users as pay per usage basis. Virtualization technology is one of the key factors in the area of cloud computing. It separates the operating system from the underlying physical machine using a hypervisor. Each virtual machine instance is having its own operating system called guest operating system. Virtualization technology provides maximum profit to the cloud providers with less maintenance cost. It allows the creation and migration of virtual machines on a physical host.

Virtualization allows the creation of multiple instances on the top of a server/host such that multiple applications can run independently. Virtual Machine (VM) is an instance which takes network, memory and I/O resources based on the user request. Figure 1 shows the framework for virtualization of physical host which is of Type-I virtualization. In Type-I virtualization, the virtual machine monitor called Hypervisor sit directly on the hardware and multiple VM's can be hosted over Hypervisor. Each VM can run multiple applications in its own operating system. Cloud provider allocates the VM to the user based on

his/her request. VMs are independent of the underlying heterogeneous Physical Machines (PM).

Live migration expands upon the thought of migration and makes it a step further. Live migration is the process of transferring Virtual Machine (VM) from one host to another without interrupting the running VMs. The "live" in live migration states that the movement should be transparent to the clients. The migrating VM should be active on source machine during the migration process. This process is useful when datacenters are facing problems like cold spots, hotspots and load unbalancing. The hotspot is the overloaded condition of a PM, where the performance of a system falls below the minimum acceptance level. The underutilization of physical machine can be defined as cold spot situation and load imbalance can occur when some machines are heavily loaded and some are lightly loaded in a datacenter.

The performance of live migration process is evaluated based on two metrics: total migration time and downtime. The duration between the start and end of the migration process is called as total migration time. The unresponsive time period of virtual machine during migration is defined as downtime. The key challenge of live migration process (Clark *et al.*, 2005) is to optimize the downtime and total migration time.

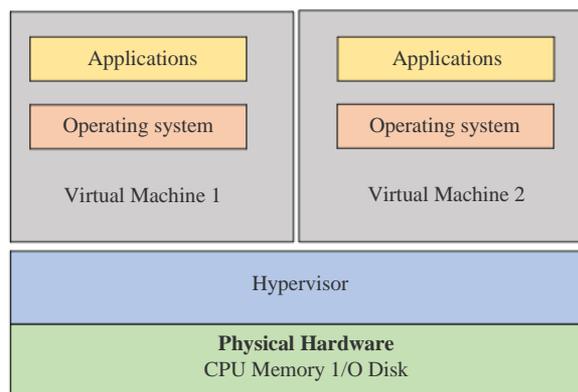


Fig. 1: Virtualization of physical node

Pre-Copy Migration

Pre-copy is a predominant memory migration technique used today by many hypervisors like VMware, Xen and KVM. It works by copying the memory of the virtual machine from the source and then transfers it to the destination iteratively without suspending the VM on the source node. In the first iteration, the whole memory is transferred from the source to destination. In the following iterations, the modified memory pages can be transmitted iteratively. This process continues until the source halts the execution of VM after transferring the last modified memory pages to the destination machine. Pre-copy (Sharma and Chawla, 2016) method consists of the following steps to migrate a VM from source to destination:

1. Initialization: The VM which is to be migrated can be selected at source machine either by direct or holistic approach
2. Reservation: This phase will check whether the destination host is having enough resources to copy the VM
3. Iterative Pre-copy: In first phase entire RAM of VM is transferred. The subsequent modified pages can be transmitted in the next iterations
4. Stop and copy: Once the pages are copied to the destination machine, suspend the VM and copy the remaining pages to destination
5. Commitment: Once a VM is copied to the destination, it will check the consistency of a VM
6. Activation: The copied VM on the destination is activated

Traditional Pre-copy method (Ma *et al.*, 2010) (Svard *et al.*, 2011), Time series approach (Hu *et al.*, 2011) Memory Compression (PhyuZaw and Thein, 2012) Memory Ballooning (Hines and Gopalan, 2009; Barham *et al.*, 2003), Working Set algorithm (Carr and Hennessy, 1981; Seok *et al.*, 2011; Zaw and Thein, 2012) are the various methods adopted to improve the

pre-copy technique. Sahni and Varma (2012) proposed a hybrid approach, which is a combination of two traditional approaches pre-copy and post copy methods. Generally, the standard migration methods like pre and post copy algorithms consume more downtime. This approach reduces the downtime by sending subset of most frequently used memory pages. This method reduces the number of page faults which occurred at post copy. An alternative way to hybrid method, (Agrawal and Pateriya, 2013) introduced live migration based on enhanced time series. This approach is done by iterative transmission of VM pages from source to destination. This algorithm suffers with iterative transmission of dirty page rate.

This paper aims to decrease the total migration time using the mirroring of VMs. The pre-copy method takes more total migration time because it has to transmit the memory pages iteratively. This method is not suitable for write intensive applications. In the proposed algorithm it is not necessary to transfer dirty memory pages iteratively to destination host because of one pass RAM copy.

In Section 2 live migration of VM based on the mirroring algorithm and its mathematical formulation is discussed. Section 4 describes the simulation environment and performance results. Finally, Section 5 includes the conclusion.

Mirroring of VMs

Cloud datacenter consists of a controller and computing nodes and Fig. 2 depicts the live migration process at datacenter using three nodes. Virtual machines are created on the top of computing nodes and can hold the resources of computing nodes. These nodes are managed by the controller node. Controller node has the authority to send messages between the computing nodes and it can directly interact with the user. The controller nodes provides the Application Programming interface (API), scheduling and other services necessary for managing the cloud resources. If computing nodes suffer with load imbalance then controller node initiates the migration of virtual machines present on computing nodes.

The mirroring algorithm can be carried out by using two layers present on the controller node: Data mover and mirror block. The first iteration of the proposed algorithm is similar to the pre-copy method. Data mover activates the one-pass RAM copy from one computing node to another computing node. It copies the files in a sequential manner to the destination node. Controller node stores the migrated VM list in linked hash table with user-id as index. When user wants to modify the memory pages at computing node, controller node verifies whether that particular page is migrated or not to the other computing node. If the page is migrated then mirror block transfers the control from one computing node to another node. Now user directly modifies the page of migrated VM present at destination computing node. Once the data mover copies all the

memory pages, VM can be suspended and activated at the destination computing node.

This approach can be considered as a modified pre-copy method where the total migration time is reduced. The following steps are carried out in the proposed method and the logical steps to migrate a VM are summarized in Fig. 3.

1. Initialization: Target VM is selected for migration based on direct or holistic approach

2. Reservation: Reserves the resources for a VM on destination node
3. Copy phase: Data mover present at the controller will copy the RAM from source to destination node
4. Mirroring phase: Mirror activation block transfers the control from source to destination whenever a user wants to modify the memory pages of already copied RAM
5. Stop and activation Phase: Once data mover copies the RAM, the VM can be removed from the source and is activated in the destination

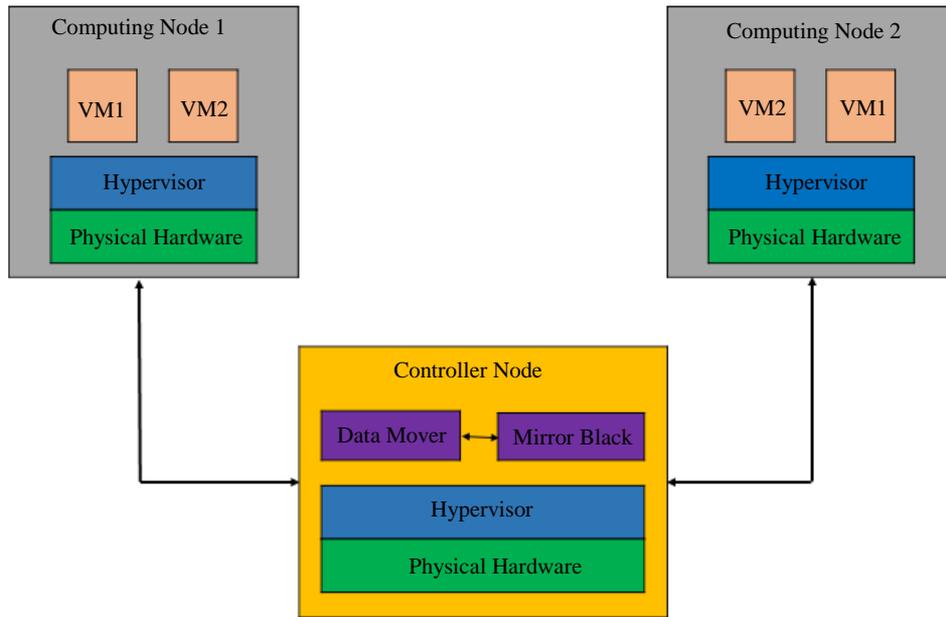


Fig. 2: Architecture diagram for live migration of virtual machines

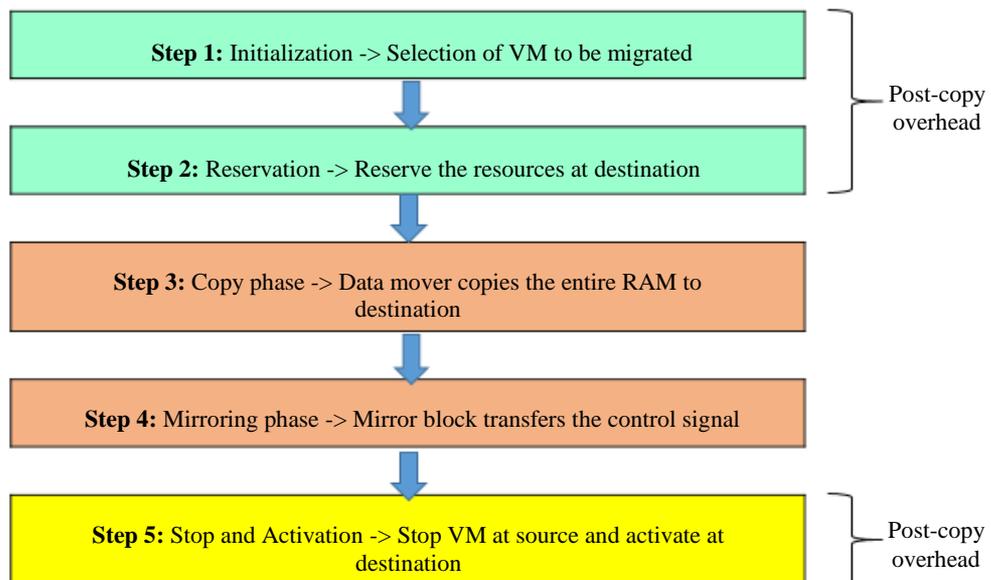


Fig. 3: Time series diagram for mirroring method

Key Parameters

The proposed method can be defined based on data transmission and control signal transmission parameters. In the traditional approach, the live migration performance is measured using the data transmission parameters like total migration time and down time. Total migration time is the time taken between the start and end of migration process. Down time is the unresponsive time of a virtual machine during migration.

Additionally, a control signal transmission parameter is used in the proposed methodology. Control transfer time is the time at which a control signal has to transfer from one computing node to another.

The time required to spent for all the above five stages is defined as the total migration time. The copy and mirroring phases dominate the total migration time due to high memory modification rate. The first two steps and the activation are considered as the pre-copy and post-copy overheads respectively.

Characterizing the Migration Performance

Migration performance (Liu *et al.*, 2011) is characterized by calculating the total migration time and the total down time. Let Cp1, Cp2 be the computing nodes holding virtual machines and Cr is a controller node that performs the live migration of VMs at Cp1 and Cp2. The VM size can be varied based on the user request can be denoted as α and the control signal size can be denoted as β . The link speed to transfer the data and control signal can be taken as δ . Let the migration of VM requires η_1 data transmission signals and η_2 control transmission signals between the controller and computing nodes.

Total migration time can be computed from the initialization to the activation phase including the copy phase. The migration of a VM depends on the size of a VM and link speed of a network. The pre and post-copy overheads are combined as overheads in the given formula. The time needed to send RAM copy along with overheads can be defined as Migration Time (MT) and is given in Equation 1:

$$\text{Migration time (MT)} = \frac{(\alpha * \eta_1)}{\delta} + \text{overheads} \quad (1)$$

In the proposed method memory of a VM transfers only once from one computing node to another and hence the migration time can be calculated using Equation 1. Down time is the duration between halting of VM at the source and its activation at the destination.

The time to transfer the control signal from source to destination can be called as control signal transmission time and is determined using Equation 2. This time is lesser than the data signal transmission time because the size of a control signal is in terms of bytes and the data transmission is in terms of Giga Bytes:

$$\text{Control Signal transmission time} = \frac{(\beta * \eta_2)}{\delta} \quad (2)$$

The Idle case of migration can be considered as the multiple read operations performed by a user without modifying the memory pages. The number of control signal transfers is equal to zero in idle case. Thus, the lower bound of control signal transfer time is equal to zero. Then again, consider the situation where all the memory pages are being changed as quickly as link speed. In this situation number of control signal transfers is equal to the number of memory pages present in a VM. Hence Equation 2 can be considered as the upper bound of control signal transfer time. The total migration time can be computed as the sum of migration and control signal transmission time and is shown in Equation 3:

$$\text{Total Migration time} = \frac{((\alpha * \eta_1) + (\beta * \eta_2))}{\delta} + \text{overheads} \quad (3)$$

In the proposed method when the VM size is constant with varying link speeds then the number of control signals is thrice of the number of data signals. Then the total migration time is shown in Equation 4:

$$\text{Total Migration time} = \frac{(\eta_1(\alpha + 3 * \beta))}{\delta} + \text{overheads} \quad (4)$$

The mirror based live migration of VMs is depicted in algorithm1. This approach will be very effective in case of write intensive applications where frequent modifications of memory pages will be done by a user. The total migration time is less in contrast to the existing methods so that the cloud provider will get maximum profit.

Materials and Methods

Traditional pre-copy and proposed mirroring algorithm has been evaluated with varying link speeds and VM sizes.

Simulation Environment

The proposed mirroring algorithm is simulated by java with respect to link speeds and VM sizes. The performance of proposed algorithm can be compared with two existing migration algorithms: pre-copy and Least Recently Used (LRU) Migration method (Zaw

and Thein, 2012). LRU method transfers the least recently used memory pages first to the destination node while migrating the VM:

Algorithm 1: Mirror based live migration of Virtual machines

```

Algorithm: Mirror based Migration ( $\rho_1, \rho_2, R$ )
 $\rho_1, \rho_2$ : Number of resources available at Cp1 and Cp2
 $\lambda$ : Threshold level to represent the resource availability at Cp1 and Cp2
R: User request like read, write
 $\theta$ : Memory (RAM) of VM at Cp1
Input:  $\rho_1, \rho_2, R$ 
Output: Migration of VM to Cp2.
begin
    1: While  $\rho_1 < \lambda$  and  $\rho_2 > \lambda$  do
        /* Cr triggers the live migration by activating data mover */
        2: transfer  $\theta$  to Cp2
        3: if  $R \rightarrow$  Read then
        4: Read  $\theta$ 
        5: end if
        6: else if  $R \rightarrow$  Write then
        7: /* Mirror Control block Activates */
        8: send control to CP2
        9: updates  $\theta$  at Cp2
        10: end while
        11: Pause VM
        12: Resume VM on Cp2
    end
    
```

Controller node and computing nodes are created in the simulation environment. Computing nodes create the VMs based on the user request and provide the resources

to them. When controller node triggers the migration process the number of data transfers can be needed to transfer the VM between the computing nodes. When user requests to update the memory pages, the computing node transfers a control signal to destination node. The number of control signal transfers required in the proposed method are more when compared to the above two approaches.

The simulation experiments were performed over the following parameter settings:

- Each Computing node holds a number of existing VMs. This count is randomly taken from 0 to 10
- The size of a VM can lies in between (64 MB to 1024 MB) and Its based on the user request
- The number of files present in the VM can vary from 10 to 100
- The link speed is assumed to be within [10 Mbps, 10 Gbps] for transmitting the data between the computing nodes

Results Analysis

Total Migration Time

The total migration time for a proposed method can be analyzed using pre-copy algorithm and LRU approach. Total migration time depends on the link speed of a network and size of a VM. Figure 4 shows the comparison among the three algorithms, where both pre-copy and LRU need more total migration time than the proposed method. As a link speed increases, the total migration time decreases drastically in all the methods. The total migration of these three algorithms is shown in Fig. 4.

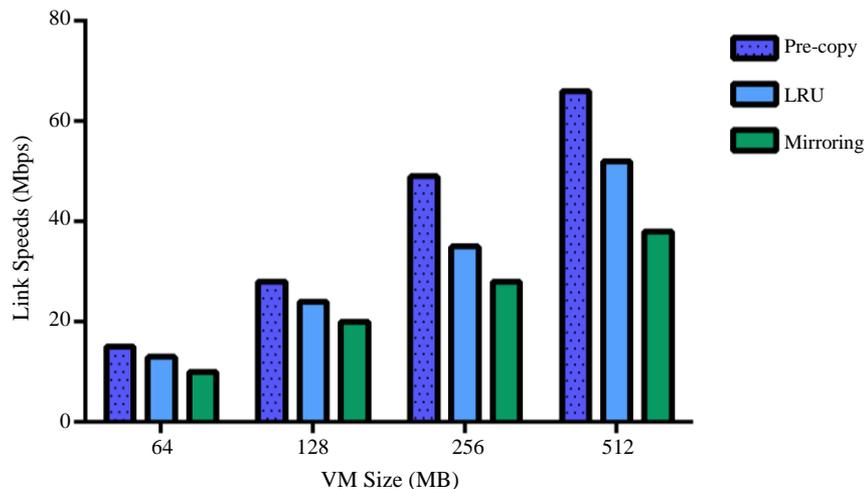


Fig 4: Comparison of total migration time

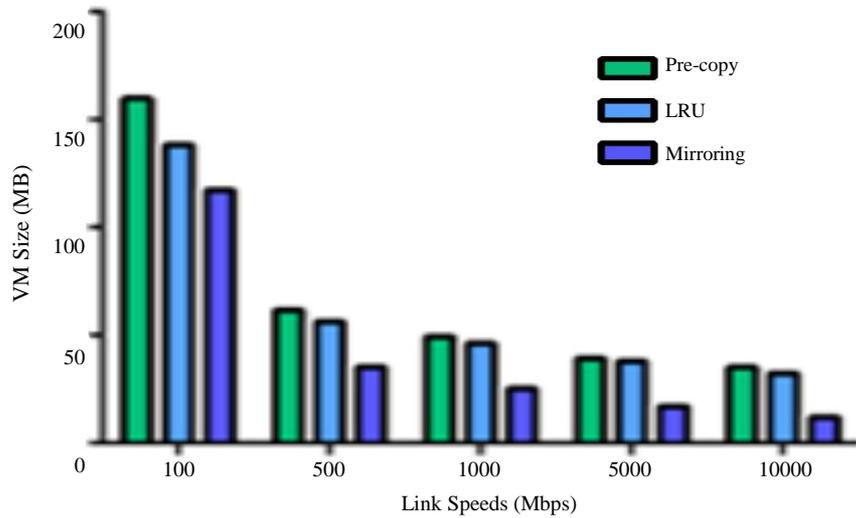


Fig 5: Comparison of Data transfer

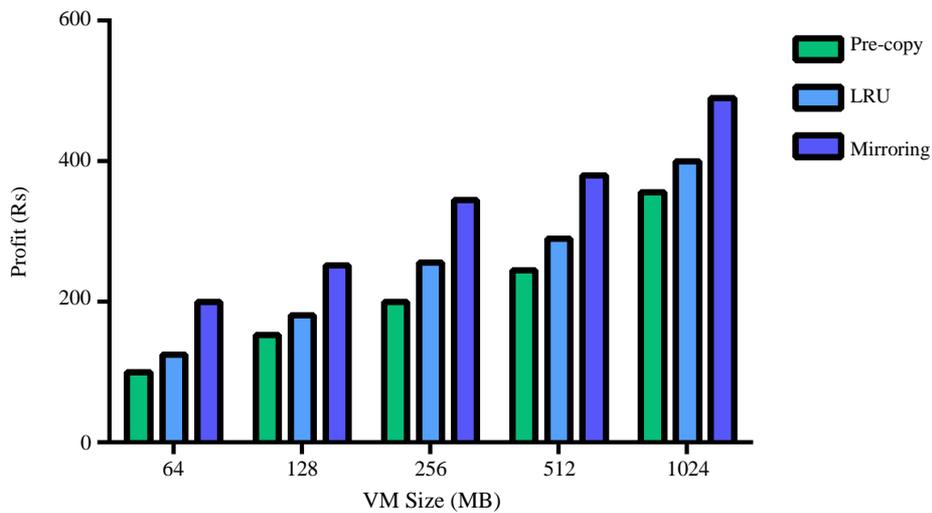


Fig. 6: Comparison of cloud Provider's profit

Table 1: Total migration time among the three methods

VM Size: 1024 GB			
Link Speed (GHz)	Pre-Copy Data: 14 Time: 1434	Mirror Data: 10 Time: 1052	LRU Data: 12 Time: 1232
100	192.360	117.400	169.880
500	77.672	35.480	71.576
1000	63.356	25.240	59.288
5000	51.867	17.048	49.457
10000	50.433	16.024	48.228

Total Data Transfer Rate

The pre-copy method iteratively transfers the modified pages between the computing nodes so the data transfer rate is higher. Figure 5 represents the number of data line transfers among three approaches. As the VM size increase

substantially, the number of data line transfers are more in pre-copy and LRU methods than the proposed algorithm.

Cloud Provider's Profit

Cloud provider offers the services as VMs, which have different resource limits with relating charges.

Figure 6 shows the produced benefits by making distinctive sizes of VMs from 64 MB to 1024 MB and Table 1 shows the snapshot results obtained using simulation. The benefits of all algorithms increment with the size of VMs. Compared with other two existing algorithms, the proposed algorithm can achieve profits by 48% and 38% respectively.

Conclusion

Live migration of virtual machines is the process of moving virtual machine from one physical host to another without interrupting other VMs. Migration of VMs supports the improved efficiency of resource usage and dynamic resource provision capabilities. The performance metrics of live migration are down time and total migration time. This paper reduces the total migration time of standard memory migration techniques by introducing the mirroring of VMs at destination host. It mainly concentrates on the Total migration time and maximum profit of providers. The proposed technique avoids the iterative transmission of memory pages from the source host to destination host. Performance results depict that the proposed algorithm can reduce 36% of the data transmitted during migration and 26% of the total migration time when compared with pre-copy migration technique. In Future we will extend the proposed technique to reduce the down time of VMs.

Acknowledgement

The authors wish to express their sincere thanks to the Department of Science & Technology, New Delhi, India project-ID:SR/FST/ETI-371/2014) and SASTRA University, Thanjavur, India for extending the infrastructural support to carry out this work. Also the authors express their thanks to Lalitha Abinaya who assisted in completing this work successfully.

Author's Contributions

All authors contributed to the final version of the manuscript.

Ethics

This article is original and contains unpublished material. The corresponding author confirms that all of the other authors have read and approved the manuscript and there are no ethical issues involved.

References

Agrawal, N. and R.K. Pateriya, 2013. Enhanced time series based pre-copy method for live migration of virtual machine. *Int. J. Adv. Eng. Technol.*, 6: 1365.

- Barham, P., B. Dragovic, K. Fraser, S. Hand and T. Harris *et al.*, 2003. Xen and the art of virtualization. *ACM SIGOPS Operat. Syst. Rev.*, 37: 164-177. DOI: 10.1145/1165389.945462
- Bhardwaj, S., L. Jain and S. Jain, 2010. Cloud computing: A study of Infrastructure As A Service (IAAS). *Int. J. Eng. Inform. Technol.*, 2: 60-63.
- Carr, R.W. and J.L. Hennessy, 1981. WSCLOCK-a simple and effective algorithm for virtual memory management. *ACM SIGOPS Operat. Syst. Rev.*, 15: 87-95. DOI: 10.1145/1067627.806596
- Clark, C., K. Fraser, S. Hand, J.G. Hansen and E. Jul *et al.*, 2005. Live migration of virtual machines. Proceedings of the 2nd Conference on Symposium on Networked Systems Design and Implementation, May 02-04, IEEE Xplore Press, CA, USA, pp: 273-286.
- Hines, M.R. and K. Gopalan, 2009. Post-copy based live virtual machine migration using adaptive pre-paging and dynamic self-ballooning. Proceedings of the ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments, Mar. 11-13, IEEE Xplore Press, NY, USA, pp: 51-60. DOI: 10.1145/1508293.1508301
- Hu, B., Z. Lei, Y. Lei, D. Xu and J. Li, 2011. A time-series based pre-copy approach for live migration of virtual machines. Proceedings of the IEEE 17th International Conference on Parallel and Distributed Systems, Dec. 7-9, IEEE Xplore Press, Tainan, Taiwan, pp: 947-952. DOI: 10.1007/978-3-319-11167-4_39
- Josep, A.D., R. Katz, A. Konwinski, L.E.E. Gunho and D. Patterson *et al.*, 2010. A view of cloud computing. *Commun. ACM*, 53: 50-58. DOI: 10.1145/1721654.1721672
- Liu, H., C.Z. Xu, H. Jin, J. Gong and X. Liao, 2011. Performance and energy modeling for live migration of virtual machines. Proceedings of the 20th International Symposium on High Performance Distributed Computing, (ISHPCD, 11), IEEE Xplore Press, pp: 171-182. DOI: 10.1145/1996130.1996154
- Ma, F., F. Liu and Z. Liu, 2010. Live virtual machine migration based on improved pre-copy approach. Proceedings of the IEEE International Conference on Software Engineering and Service Sciences, Jul. 16-18, IEEE Xplore Press, Beijing, China, pp: 230-233. DOI: 10.1109/ICSESS.2010.5552416
- PhyuZaw, E. and N.L. Thein, 2012. Improved live virtual migration using LRU and SPALY tree algorithm. *Int. J. Comput. Sci. Telecommun.*
- Sahni, S. and V. Varma, 2012. A hybrid approach to live migration of virtual machines. Proceedings of the IEEE International Conference on Cloud Computing in Emerging Markets, Oct. 11-12, IEEE Xplore Press, Bangalore, India, pp: 1-5. DOI: 10.1109/CCEM.2012.6354587

- Seok, H., Y. Park, K.W. Park and K.H. Park, 2011. Efficient page caching algorithm with prediction and migration for a hybrid main memory. ACM SIGAPP Applied Comput. Rev., 11: 38-48. DOI: 10.1145/2107756.2107760
- Sharma, S. and M. Chawla, 2016. A three phase optimization method for pre-copy based VM live migration. Springer Plus, 5: 1022. DOI: 10.1186/s40064-016-2642-2
- Svard, P., J. Tordsson, B. Hudzia and E. Elmroth, 2011. High performance live migration through dynamic page transfer reordering and compression. Proceedings of the IEEE 3rd International Conference on Cloud Computing Technology and Science, Nov. 29-Dec. 1, IEEE Xplore Press, Athens, Greece, pp: 542-548. DOI: 10.1109/CloudCom.2011.82
- Zaw, E.P. and N.L. Thein, 2012. Improved live VM migration using LRU and splay tree algorithm. Int. J. Comput. Sci. Telecommun., 3: 1-7.