Journal of AI and Data Mining
Vol 7, No 4, 2019, 589-596
DOI: 10.22044/JADM.2018.7178.1847

Analytical Evaluation of an Innovative Decision-making Algorithm for VM Live Migration

M. Tajamolian and M. Ghasemzadeh*

Computer Engineering Department, Yazd University, Yazd, Iran.

Received 18 June 2018; Revised 11 August 2018; Accepted 06 October 2018
**Corresponding author: m.ghasemzadeh@yazd.ac.ir (Ghasemzadeh).*

Abstract
In order to achieve the virtual machine live migration, the two "pre-copy" and "post-copy" strategies are presented. Each one of these strategies, depending on the operating conditions of the machine, may perform better than the other. In this article, a new algorithm is presented that automatically decides how the virtual machine live migration takes place. In this approach, the virtual machine memory is considered as an informational object that has a revision number and it is constantly changing. We determine the precise criteria for evaluating the behavior of a virtual machine and automatically select the appropriate live migration strategy. Also different aspects of the required simulations and implementations are considered. Analytical evaluation shows that using the proposed scheme and the presented algorithm can significantly improve the virtual machine live migration process.

Keywords: *Virtual Machine Live Migration, Pre-copy and Post-copy, Quadruple Adaptive Version Numbering Scheme, Decision-making Algorithm, Cloud Computing.*

1. Introduction

In the recent years, advances in cloud computing based on the idea of "pay as you go" have been a major influence in development of IT [1]. The Virtual Machine (VM) is one of the most major and infrastructural technologies in cloud computing. Among the key benefits of VMs, the ability of live migration of virtual machines is significant. So far, several methods have been proposed for memory live migration in virtual machines. Each method proposed by different researchers has strengths and weaknesses (pros. & cons.) versus the others. Due to the volatile workload conditions of VMs, it is possible that a particular VM live migration method that has had acceptable performance in some circumstances has not shown good results in some other situations.

Based on the concept of the "Informational Object" and a "Quadruple Adaptive Version Numbering Scheme" for tracking the informational object changes, which was first presented by us in another paper [2], we have tried a new approach to the issue of choosing the best method for VM live migration. Naturally, to propose a decision-making algorithm, we need some criteria. Thus, we use the QAVNS-based fields to provide our criteria. Then based on the criteria, an automated decision-making algorithm is proposed.

Using our proposed algorithm in the de facto hypervisors enables CDC (Cloud Data Center) administrators to rely on the automated VM live migration method selection. This improves the overall performance of CDCs and decreases probability of VM live migration faults.

The remainder of this article is organized as what follows. In the next section, some basic concepts are mentioned. In the third section, we will review the research background in the field of virtual machine memory live migration. In Section 4, our proposed approach is presented. In Section 5, we describe the implementation of the proposed approach and provide an algorithm. In Section 6, the analysis of the proposed algorithm is discussed. At the end (in Section 7), conclusions and suggestions for the future works are made.

2. Background

A virtual machine is, in fact, a software implementation of a real physical machine that runs on a physical machine and behaves like that. The software layer that performs the simulation of the virtual machine is called the Hypervisor or Virtual Machine Monitor (VMM).

The term "Virtual Machine Migration" means that a snapshot of an operating system running on a virtual machine, along with all process images, system state, and state of memory, network connections, storage devices, and other related items from a hardware platform (known as a host machine) are moved to another hardware platform. Migration has a similar procedure to the virtual machine suspension, except that instead of storing the virtual machine state in the state image files, this information is transferred via the network from the origin hardware machine to another hardware machine (named destination), and the task of restoring the virtual machine occurs on that machine instead of the origin machine. The term "Live Migration" means that the chosen virtual machine for migration will be transferred from a hardware machine to another without being shutdown or even suspended for a long time. Thus at the same time, as the point of view of a user, it sounds that the VM is active and responsive to the received requests.

The most ordinary way for transferring a virtual machine from a hardware platform to another one without full stop of its guest operating system is to divide the procedure into three phases: suspend, copy, and restore. Some researchers have called this method the "Naive Method" [3]. In the recent studies, the procedure of virtual machine migration is divided into three phases: push, suspend & copying, and pull [4].

3. Literature review

One of the proposed methods is "pure demand-migration", which was presented many years ago by Zayas [5]. Of course, Zayas's research work focused on the issue of "process migration", which somehow shaped the topic of research on "virtual machine live migration" in the next two decades.

The most popular and most used method is the so-called "pre-copy", proposed by Clark et al. [4]. In this method, in the n-th iteration, those pages of memory that have changed in the (n-1)-th iteration are transferred to the destination machine. The pages of memory that change so fast and continuously that can not be transferred by iterative push are called "Writeable Working Set" or WWS [4]. The push stage is iterated as long as WWS is reduced to a significant degree or that the number of iterations of this stage exceeds a pre-determined limit. One of the implementations is Quasar, which has used this method to reduce the service suspend time [6].

Hu and colleagues have proposed a method that works based on the behavior of access to the virtual machine memory. It defers placing pages of memory that are likely to change in the WWS, which is transferred in the push phase until the last iteration [7]. Another algorithm has been presented by Khalid Ibrahim of the Berkeley Lawrence National Laboratory, which partly reduces the problem of duplicate transferring of the memory pages in the pre-copy but does not fix it [8].

In order to complete the post-copy method, four techniques were proposed that differentiated them from the "pure demand-migration" approach provided by Zayas: Demand-paging, Active Push, Prepaging, and Dynamic Self-Ballooning [9].

Unfortunately, despite the undeniable similarity of the Hines and colleagues' method in their article (named post-copy) to the one proposed by Zayas, they have not mentioned anything to him in their article! We think that this can be considered as a kind of plagiarism and should be blamed and criticized.

Of course, along with the above, techniques have also been proposed to compress memory pages to reduce network traffic and increase performance of migration [10-12]. As the latest advancement in this field, we can refer to a combinational approach of pre-copy and post-copy approaches suggested by Sahni and colleagues [13].

4. Approach and proposed method

In order to provide an automated algorithm for choosing a virtual machine live migration method, we provide criteria for automatically detecting virtual machine state. Based on the concept of the "Informational Object" presented in [2], we assume virtual machine memory as an informational object that is constantly changing and can therefore have a "QAVNS Revision Number" at any moment.

Assume that the *MST* (Migration Start Time) constant represents the start time of the live migration of virtual machine memory.

$$jiffies = count\,(TimerInterrupts) \quad (1)$$

$$MST = jiffies\ at\ start\ of\ migration\ task \quad (2)$$

The suggested mapping of QAVNS fields in the VM memory informational object is as follows:

The *0RSN* field in the QAVNS scheme is known as the "Release Sequence Number". Given the nature of the virtual machine memory informational object, we consider this field as a sequence number

that has a time nature and its values are extracted from machine timer interrupts. This field represents the number of moments that have elapsed since the start of the migration operation. Therefore, the *0RSN* field of the QAVNS scheme is mapped here as the "jiffies" with the *JS* acronym, and is supplied according to the below flowchart.

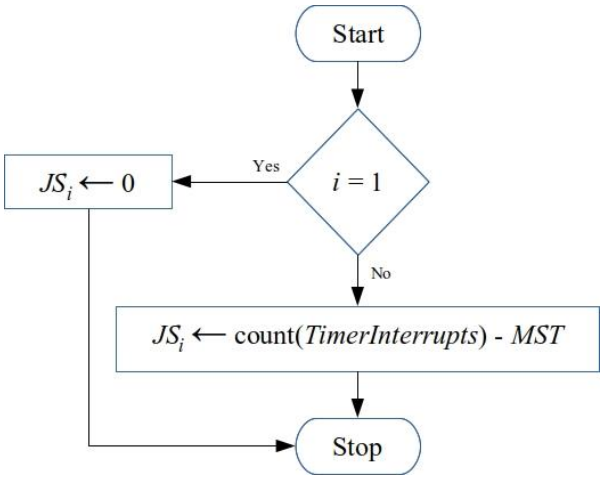

Figure 1. Supply appropriate values for the *JS* field.

The *GG* field in the QAVNS scheme is known as the "Generation number". Given the nature of the virtual machine memory informational object, we consider this field to be the "Generation Number of the virtual machine operating system". Therefore, the *GG* field of the QAVNS scheme is mapped here as the "Operating System Instance Number" with the *OSIN* acronym, and is supplied according to the below flowchart:

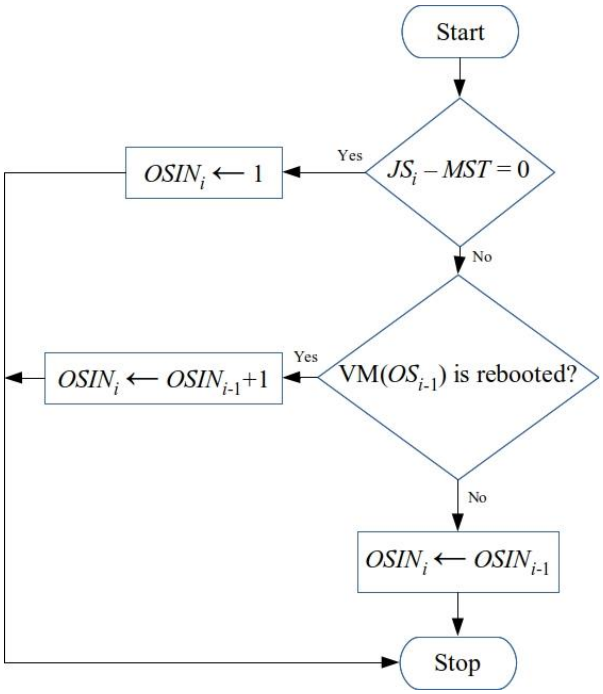

Figure 2. Supply appropriate values for the *OSIN* field.

The *FF* field in the QAVNS scheme is known as the "Feature List Number". Given the nature of the virtual machine memory informational object, we consider this field to be an "indicator of the volume of changes in processes of the virtual machine". Therefore, the *FF* field of the QAVNS scheme is mapped here as the "Process Change Number" with the *PCN* acronym, and is supplied according to the below flowchart.

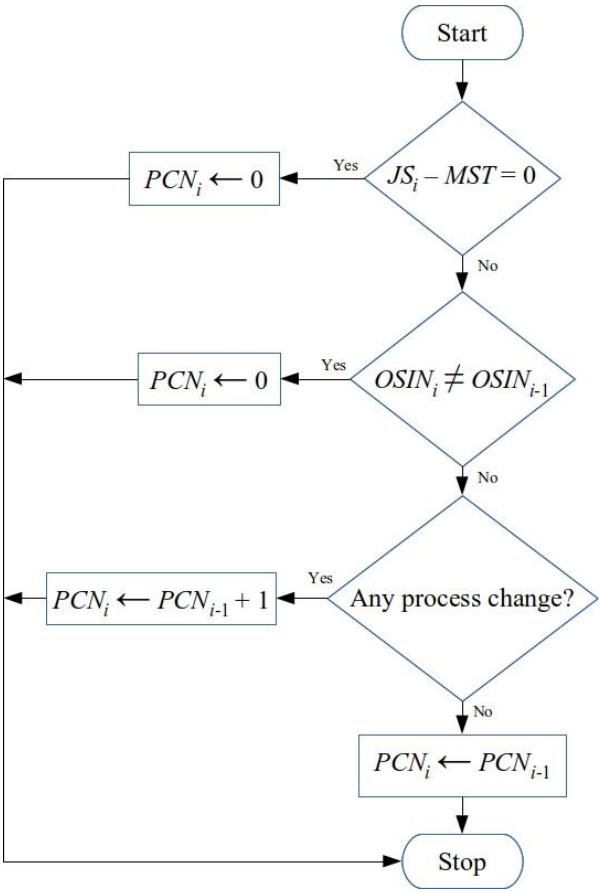

Figure 3. Supply appropriate values for the *PCN* field.

The *CCC* field in the QAVNS scheme is known as the "Correction List Number". Given the nature of the virtual machine memory informational object, we consider this field to be "number of the modified memory pages". Therefore, the *CCC* field of the QAVNS scheme is mapped here as "Dirty Page Count Number" with the *DPCN* acronym, and is supplied according to the Figure 4.
Therefore, the "Version Identifier Scheme" for virtual machine memory as a QAVNS-based informational object will be:

$$JS.OSIN.PCN.DPCN \qquad (3)$$

We add an initial step called "Sampling Phase" to the three phases that Clark and colleagues proposed [4] for the virtual machine migration process. During this phase, the values for the virtual machine memory version identifier are updated and monitored. Determining the "length of the sampling phase" (which we call it T_s) is still an open research topic.

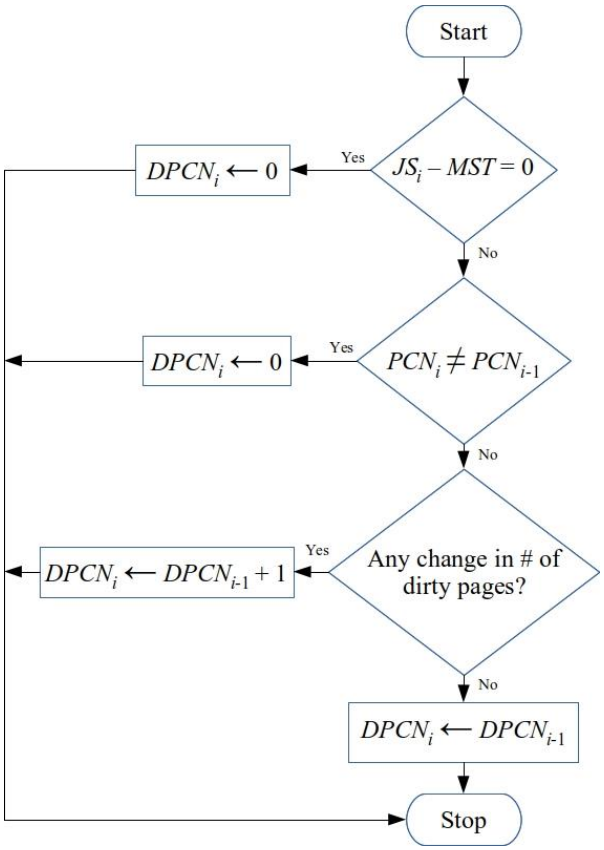

Figure 4. Supply appropriate values for the *DPCN* field.

5. Algorithm and implementation

Simulation and implementation can be used to provide evidence of the efficiency of the proposed method.

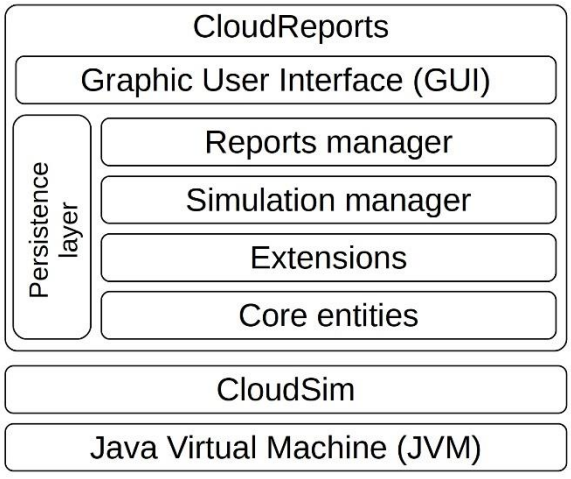

Figure 5. CloudReports Architecture and its layers [14].

Bahwaireth et al., in their research work, have pointed to some of the most promising Cloud Computing simulation tools for CloudSim, CloudReports, CloudExp, iCanCloud, CloudAnalyst, and GreenCloud [14]. Of course, some other items such as TeachCloud and GroudSim can also be added to the list.

Many other tools (including CloudReports) have been developed based on CloudSim. In Figure 5, the CloudReports architecture is shown as is described in the paper by Bahwaireth et al.

Based on the configuration shown in Table 1, an experiment was conducted on CloudAnalyst. The table describes the user base and the data centers used.

Table 1. Configuration of the user base and the data center used by CloudAnalyst [14].

Name	Region	Requests per User per Hr	Data Size per Req (bytes)	Peak Hours Start (GMT)	Peak Hours End (GMT)	Avg Peak Users	Avg Off-Peak Users
UB1	0	60	100	13	15	400000	40000
UB2	1	60	100	15	17	100000	10000
UB3	2	60	100	20	22	300000	30000
UB4	3	60	100	1	3	150000	15000
UB5	4	60	100	21	23	50000	5000

Data Center	#VMs	Image Size	Memory	BW
DC1	20	10000	1024	1000
DC2	30	10000	1024	1000
DC3	40	10000	1024	1000

This test leads to the following results:

Table 2. Results from simulation using CloudAnalyst [14].

Data Center	AVG (ms)	Min (ms)	Max (ms)
DC1	69.62	1.94	168.53
DC2	61.58	33.47	81.84
DC3	30.27	12.50	50.96

According to the authors, tables 1 and 2 are illustrative screenshots of a request servicing time for each data center and cost details, respectively". Finally, Bahwaireth and colleagues in their article introduced "iCanCloud" as the top choice.

Tian and his colleagues compared four open-source simulation tools in this area, namely CloudSim, GreenCloud, iCanCloud, and CloudSched [15]. Their research work gives the results shown in table 3.

Additionally, Hirofuchi and colleagues claimed that the SimGrid toolkit as a commonly used simulation tool for distributed systems research can be extended to simulate the live migration of virtual machines. According to them, this developed tool has the ability to simulate sophisticated models containing several hundred thousand machines on thousands of physical machines [3].

Table 3. Comparison of the characteristics of the four simulation tools [15].

Items	Cloud-Sim	Green-Cloud	iCan-Cloud	Cloud-Sched
Platform	Any	NS2	OMNET , MPI	Any
Programming language	Java	C++/ OTcl	C++	Java
Availability	Open source	Open source	Open source	Open source
Graphical support	Limited (Via CloudAnalyst)	N	N	Y
Physical models	N	Limited (Via Plug-in)	Y	Y
Models for public cloud	N	N	Y	Y
Parallel experiments	N	N	Y	N
Energy consumption	Y	Y	N	Y
Migration algorithms	Y	N	N	Y
Simulation time	Seconds	Tens of minutes	Seconds	Seconds
Memory space	Small	Large	Medium	Small

In figure 6, the graph of the correlation between the "CPU Utilization" and "Memory Update Speed", as reported by Hirofuchi and colleagues, is illustrated by the workload generated by PostgreSQL database software:

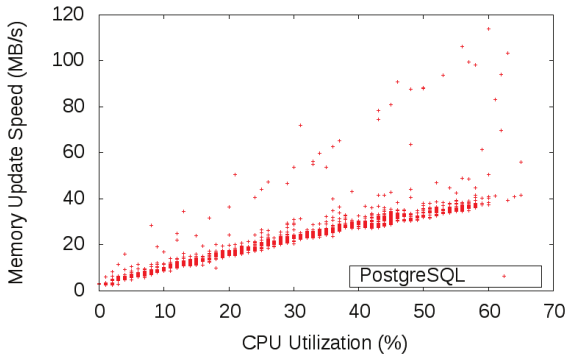

Figure *6*. The degree of correlation between the "CPU Utilization" and the "Memory Update Speed" [3].

Based on this graph, they set the value of α in the equation $Y = \alpha X$ with an approximation of 60 MB/s. In the experiments conducted in real-world using simulation tools developed on SimGrid by Hirofuchi and colleagues, virtual machines have 2 GB of memory, 90% of which are intended as Working Set.

The memory update intensity is also assumed to be similar to PostgreSQL database behavior (60MB/s when used with 100% of CPU capacity).

The results are shown in Figure 7.

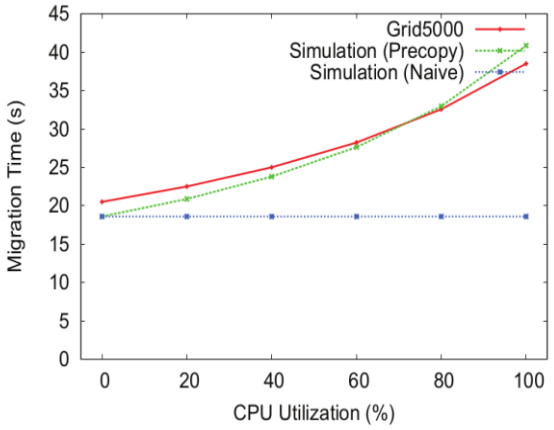

Figure *7*. Comparison of Real Migration Time and Simulation [3].

As shown in figure 4, the results of the simulation using the development carried out on SimGrid are promising compared to the results of the migration in the real-world.

The virtual machine live migration algorithm in CloudSim [16] is calculated based on the following formula:

$$TLoM = \frac{mem\_size(VM)}{bandwidth(Net)} \quad (4)$$

In that sense, *TLoM* is the "Time Length of virtual machine Migration". According to Hirofuchi and his colleagues, who are also endorsed by us, this calculation can not provide the most realistic results for most of the time due to the virtual machine memory modification conditions. They also believed that adding capability of a logical simulation of live migration to CloudSim required a large amount of programming and change in its source code [3].

Li and colleagues have proposed another solution. They have implemented a tool called "DartCSim+" based on CloudSim, and have been able to fix the three CloudSim limitations in their proposed tool [17]: first is the lack of simultaneous support for power and network models; the second is the lack of power-based simulation support for network components; and the third is not considering network overhead on the live migration process of virtual machines. In Figure 8, the results of one of the simulations of Li and his colleagues are shown using DartCSim+, which confirm that with increase in the intervals between migrations, the rate of retransferring of packets drop rapidly.

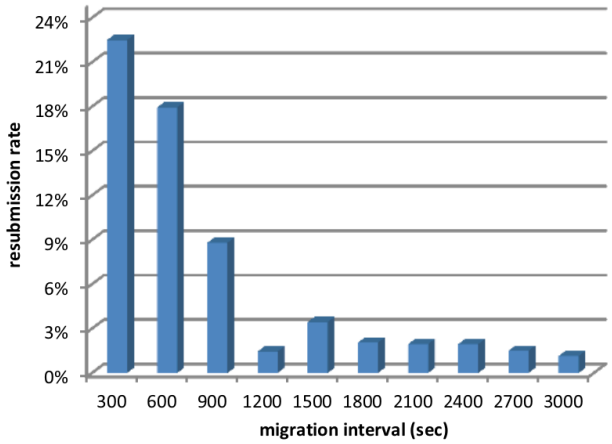

Figure 8. Reduction of packet resubmission rates by increasing the migration intervals [17].

The initial results from our studies are not consistent with the results of the research work with Bahwaireth et al. [14], i.e. iCanCloud, and so we are not in a position to confirm their research results. The preliminary results of our studies indicate that CloudSim is superior to other options in terms of capabilities, and can be a good choice for simulating the various VM live migration methods. In addition to the results of Li et al. [17], the results of the research work by Stamenov et al. also confirm the initial results of our investigations [18]. We also believe that SimGrid [3] and DartCSim+ [17] can be good alternatives to CloudSim in some circumstances.

The VM memory revision identifier is displayed at the start of the sampling phase with the index s and at the end of the sampling phase with the index e. Thus we will have:

Revision identifier at the beginning of the sampling phase:

$$JS_s.OSIN_s.PCN_s.DPCN_s \quad (5)$$

Revision identifier at the end of the sampling phase:

$$JS_e.OSIN_e.PCN_e.DPCN_e \quad (6)$$

We claim that the following algorithm can automatically detect the different operating conditions of a virtual machine and choose the appropriate approach for live migration appropriately. The flowchart of the proposed method is as below:

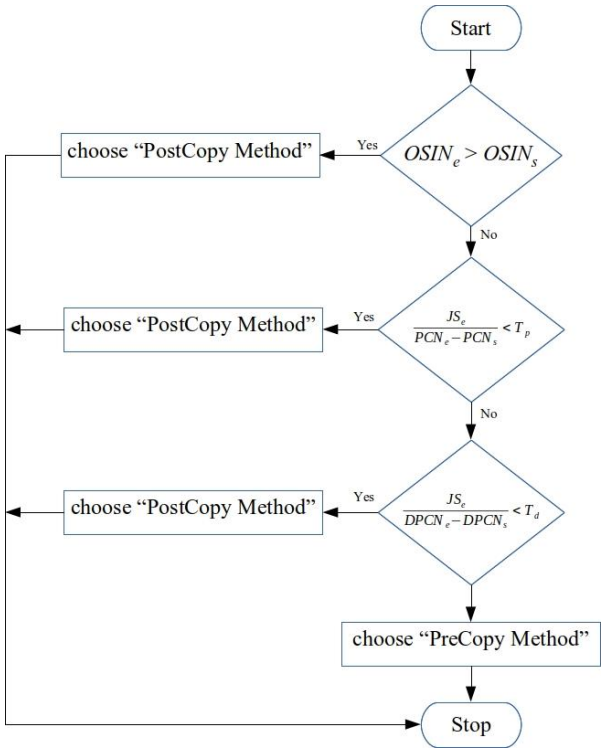

Figure 9. Detecting the different operating conditions of a virtual machine.

Determining appropriate values for the T_p and T_d threshold constants in the above flowchart is still an open research topic.

6. Analysis

Based on the research works, it has been determined that each one of the pre-copy and post-copy approaches is more appropriate than the other in certain conditions, and vice versa. In short, the use of the pre-copy method provides a better performance when the processes running on the virtual machine show more "Read-intensive" behaviors. To the contrary, using the post-copy method is more efficient at the conditions when running processes on a virtual machine are more "Write-intensive" [9]. It needs to be noted that research work has been carried out on both pre-copy and post-copy methods after their initial presentation, and suggestions have been made to complete/improve them. Thus when referring to the pre-copy and/or the post-copy method, we mean the initial proposed method and its entire improved species.

If during the sampling phase, by increasing the value of the *OSIN* field, it is determined that the virtual machine operating system is being restarted, this analysis can be deduced that the volume of changes to the memory pages is large and incremental. This volume of changes is "Level One" (means the highest level of changes in memory pages), and yet it is no longer necessary to

examine other virtual machine working states for choosing the appropriate method. We call this state the "Unstable state of the operating system of the VM", and therefore, the appropriate choice in this case is the post-copy.

If there is no change in the *OSIN* field, then in the next step, it is possible to decide based on the rate of change in the process set that is managed by the operating system (based on changes in the *PCN* field) in a time unit. A logical inference here is that, typically, by creating or deleting any process in a virtual machine, the volume of changes in memory pages is more than the normal operation.

If the resulting change rate of the process set that is managed by the operating system passes through a threshold (which in the previous section shown with a T_p constant), this volume of changes is "Level Two" (means the medium level of changes in memory pages); however, it is no longer necessary to examine other virtual machine conditions to choose the appropriate method. We call this state a "Storm of birth and death of processes in a virtual machine, and therefore, the appropriate choice in this case is the post-copy.

If none of the above two events occur, then it is time to decide based on the third criterion, which is the rate of the number of modified memory pages (based on changes in the *DPCN* field) per unit time. If the rate of increase in the number of modified memory pages passes through a threshold (which is shown in the previous section with a T_d constant), this volume of changes is the "Third Level" (means the lowest noticeable level of changes in memory pages). This is a simple, yet quite reasonable deduction, that it would be a good choice for the post-copy method. The occurrence of this state indicates that while the virtual machine system is in a steady state and also does not see the storm of birth and death of processes, we are countering with a number of write-intensive processes that are constantly changing the memory pages. We call this an "Unruliness of processes", and therefore, the appropriate choice in this case will be the post-copy method. Obviously, if none of the above three cases are true for a virtual machine, the amount of changes in memory pages is low. We call this a "Calm virtual machine with noble processes", and so the proper choice in this case will be the pre-copy.

7. Conclusions and suggestions

This paper presents a new approach for automating the process of choosing the right strategy for live migration of virtual machine memory and state. The proposed approach can enable systems based on virtual machine technology and cloud computing to automatically select the best strategy and VM migration algorithm in constantly changing conditions. This leads to a relative optimum performance in the current situation. One of the most significant concerns in this research work is: whether the overhead of the proposed algorithm outweighs the benefits of it or not? To answer this question, it is necessary to carry out the exact simulation or actual implementation. In line with this work, finding optimal values for the three parameters T_s , T_p , and T_d is very valuable and influential.

References

[1] Rezvani, M. (2018). Assessment Methodology for Anomaly-Based Intrusion Detection in Cloud Computing, J. of AI and DM (JAIDM), vol. 6, no. 2, pp. 387–397, Jul. 2018.

[2] Tajamolian, M. & Ghasemzadeh, M. (2018). QAVNS: An adaptive version numbering scheme for informational objects used in virtual machines live migration, 4th Nat. Conf. on Distributed Computing and Big Data Processing, Tabriz, Iran, 2018.

[3] Hirofuchi, T., et al. (2013). Adding a Live Migration Model into SimGrid, Int. Conf. on Cloud Computing Tech. and Science, Washington, USA, 2013, vol. 1, pp. 96–103.

[4] Clark, C., et al., (2005). Live migration of virtual machines, 2nd Symp. Networked Systems Design & Implementation, Boston, 2005, vol. 2, pp. 273–286.

[5] Zayas, E. (1987). Attacking the process migration bottleneck, ACM SIGOPS Operating Sys. Review, vol. 21, no. 5, pp. 13–24, 1987.

[6] Venkatesha, S., et al. (2009). Survey of virtual machine migration techniques. Technical report, Dept. of Computer Science, University of California, Santa Barbara, 2009.

[7] Hu, B., et al. (2011). A time-series based pre-copy approach for live migration of virtual machines, IEEE 17th Int. Conf. Parallel and Distributed Systems, Tainan, 2011, pp. 947–952.

[8] Ibrahim, K. Z., et al. (2011). Optimized pre-copy live migration for memory intensive applications, Int. Conf. for High Performance Computing, Networking, Storage and Analysis, Washington, 2011, pp. 40–50.

[9] Hines, M. R., et al. (2009). Post-copy live migration of virtual machines, ACM SIGOPS OS review, vol. 43, no. 3, pp. 14–26, 2009.

[10] Jin, H., et al. (2009). Live virtual machine migration with adaptive memory compression, IEEE Int. Conf. Cluster Computing and Workshops, CLUSTER'09, LA, USA, 2009, pp. 1–10.

[11] Svärd, P., et al. (2011). High performance live migration through dynamic page transfer reordering and

compression, Int. Conf. of Cloud Computing Technology & Science, Athens, Greece, 2011, pp. 542–548.

[12] Svärd, P., et al. (2011). Evaluation of Delta Compression Techniques for Efficient Live Migration of Large Virtual Machines, 7th ACM SIGPLAN SIGOPS Int. Conf. on Virtual Execution Environments, Newport Beach, USA, 2011, vol. 46, no. 7, July 2011, pp. 111–120.

[13] Sahni, S. & Varma, V. (2012). A hybrid approach to live migration of virtual machines, Int. Conf. Cloud Computing in Emerging Markets, Bangalore, India, 2012, pp. 1–5.

[14] Bahwaireth, K., et al. (2016). Experimental comparison of simulation tools for efficient cloud and mobile cloud computing applications, EURASIP J. on Info. Security, vol. 2016, no. 1, p. 14, Dec. 2016.

[15] Tian, W., et al. (2015). Open-source simulators for Cloud computing: Comparative study and challenging issues, Simulation Modelling Practice and Theory, vol. 58, no. Part 2, pp. 239–254, Nov. 2015.

[16] Calheiros, R. N., et al. (2011). CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms, J. of Software: Practice and Experience, vol. 41, no. 1, pp. 23–50, Jan. 2011.

[17] Li, X., et al. (2013). DartCSim+: Enhanced CloudSim with the Power and Network Models Integrated, IEEE Int. Conf. on Cloud Computing, Santa Clara, USA, 2013, pp. 644–651.

[18] Stamenov, D. & Kostoska, (2017). Virtual machine migration in Cloud–techniques, challenges and CloudSim migration simulation, 14th Int. Conf. on Informatics and Inform. Technologies, Mavrovo, Macedonia, 2017, pp. 103–109.