Svedberg Open
DISSEMINATION OF KNOWLEDGE

International Journal of Artificial Intelligence and Machine Learning

Publisher's Home Page: https://www.svedbergopen.com/

Research Paper Open Access

An Adaptive Pre-Copy Live Virtual Machine Migration Framework Using Fire Hawks Optimization Algorithm for Efficient Cloud Resource Management

A.Nagaswathy[1*],Dr. M.Suganya[2]

[1]Research Scholar, Department of Computer Science,Rathinavel Subramaniam College of Arts and Science,Coimbatore
Email :nagaswathy13@gmail.com, https://orcid.org/0000-0001-9654-4843

[2]Associate Professor &HoD,School of Computer Studies(UG),Rathinavel Subramaniam College of Arts and Science,Coimbatore
Email : suganya@rvsgroup.com, https://orcid.org/0000-0002-5572-3236

Abstract

The Live Migration of Virtual Machine (VM) is essential in cloud computing to ensure the service is not interrupted. This becomes especially necessary when performing system maintenance, load balancing, or resource management-related operations. One of the most popular migration strategies is the pre-copy, in which the virtual machine (VM) continues to operate while its memory contents are transferred to the target host. Nonetheless, this process can be quite inefficient depending on the Dirty Rate (DR), this is the number of pages of memory that are changed while the migrating is occurring. An inappropriate DR may lead to longer migration time, longer down time and ineffective resource usage. To tackle these challenges, this research paper proposes an optimized Pre-Copy Live VM Migration (PLVM) technique that adopts the Fire Hawks Optimization (FHO) algorithm to adaptively select the best Dirty Rate during migration. Total migration time, downtime, and resource are used to create a fitness function utilization to optimize performance for migration. The process of optimization seeks to reduce migration overhead and enhance effective utilization of computational resources. The FHO-based PLVM technique is implemented using CloudSim simulation environment and evaluated using NASA workload traces to represent real user request pattern. The proposed method's effectiveness is contrasted with that of current migration methods, which are Machine Learning-based PLVM (ML-PLVM) and OMA-based PLVM (OMA-PLVM). Test results show that the method reduced downtime and migration time while significantly improving CPU and memory utilization. The results indicate that the efficiency of live VM migration in cloud data centres may be enhanced by the suggested optimization strategy.

Keywords: *Live Virtual Machine Migration, Pre-copy Live VM Migration (PLVM), Fire Hawks Optimization Algorithm, Dirty Rate Optimization, Cloud Computing, CloudSim, Migration Downtime Reduction, Resource Utilization.*

1. Introduction

Cloud computing is the provision of scalable and on-demand computing services over the internet, according to the US National Institute of Standards and Technology [1]. This allows businesses to utilize and pay for only the resources they need without investing in actual physical structures. As cloud-based applications rapidly grow, Service availability and Resource management of cloud Service providers become a major concern [2]. By enabling several Virtual Machines (VMs) to run on a single machine while effectively sharing hardware resources, virtualization technology helps ameliorate these requirements[3].

A key component for efficient resource management in cloud environments is live virtual machine migration. It is possible to move a running VM from one physical machine to another without affecting the functioning of the applications [4,5,6]. The ability of cloud providers to allow dynamic relocation of VMs enhances resource utilization and guarantees that service delivery remains uninterrupted [7].

The most popular migration method is pre-copy. While a VM is functioning, its memory is progressively transferred from a source host to a destination host. While this procedure is carried out, memory pages that

have been altered are categorized as dirty pages, which must be transferred again [8,9]. The iteration occurs until the amount of data that is modified becomes sufficiently small for the VM will be briefly suspended and it can resume on the destination server. Though this procedure ensures service continuity, the effectiveness of the migration mechanism predominantly depends on several parameters involved in the migration mechanism [10].

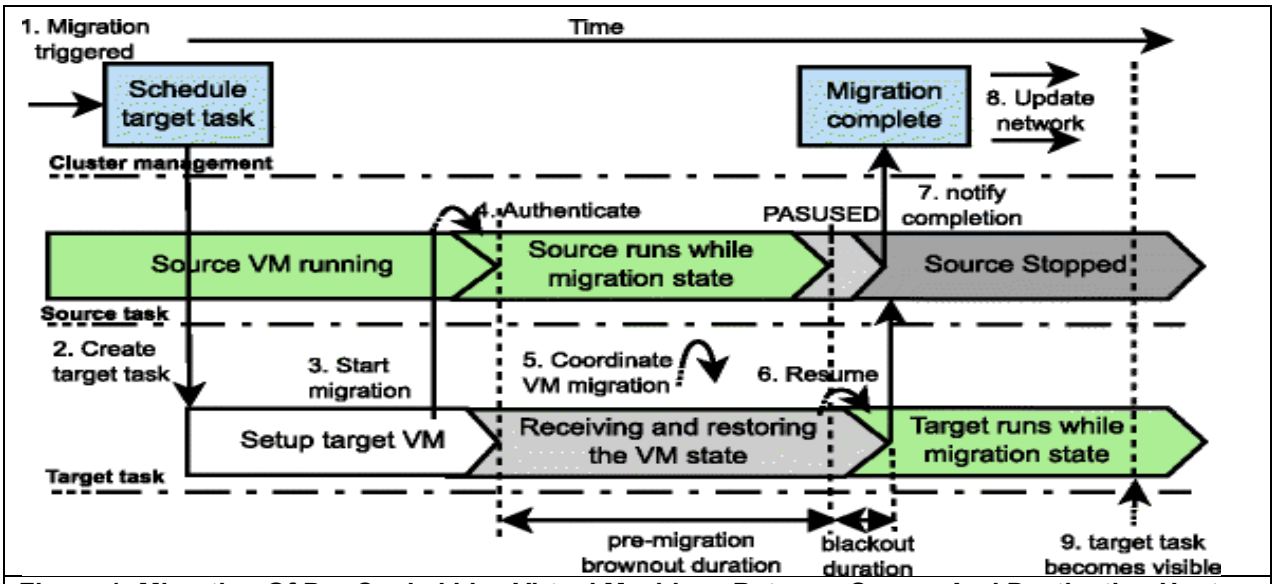

Figure 1: Migration Of Pre-Copied Live Virtual Machines Between Source And Destination Hosts

Figure 1 depicts the architecture of the live VM migration based on the cloud computing environment.Without interrupting the performance of the program, the migration procedure transfers a running VM states from the source host to the desired host across a network.While the VM continues to operate on the source system, the pre-copy method entails regularly copying the memory pages. As less data remains, the dirty pages are transmitted repeatedly. Ultimately, the VM is briefly put on hold before being resumed at the destination host which finalizes the migration process [11].

According to [12,13], a crucial component of cloud data centers is now the live VM migration. It helps in server consolidation, load balancing, energy-efficient resource allocation and other operational objectives. During hardware upgrades and system maintenance, VM applications could be migrated between servers without shutting down VM applications. In similar cases, the underutilized VMs of certain physical machines are migrated to other servers [14], which allows idle machines to be turned off to save energy. These capabilities ultimately enhance the performance and reliability of cloud infrastructures.
Nonetheless, certain performance issues are raised in the migration. There are various reasons why the total time taken to provide service during migration and the downtime duration might get affected. Network bandwidth, memory size, characteristics of workload and rate of updating memory while performing migration, etc. are such reasons. The Dirty Rate (DR) measures memory page modification during migration. The DR has a significant impact on efficiency among these other attributes. If the sampling rate is high, then the system should have to repeatedly transfer modified memory pages. And, that can increase migration iterations and prolong the overall migration [15]. As a result, choosing a suitable dirty rate is essential to making migration overhead low that can enhance the efficiency of live VM migration.

1.1. Problem Statement

Despite the popularity of pre-copy live migration thanks to their service continuity; it still faces challenges above migration overhead and resource efficiency issues. The overall downtime and migration time are significantly increased by poor choice of migration parameters especially dirty rate. This can also cause wastage of computational resources in cloud data centres.

To solution these issue, it is desirable to develop an optimal migration strategy which can adaptively find suitable migration parameters through the exploitation of system condition. In this study, a new optimized Pre-copy Live VM Migration (PLVM) technique is proposed in which Fire Hawks Optimization (FHO) algorithm is employed to determine and select the optimal dirty rate during migration. The method under consideration will enhance migration time, downtime, and resource use. The proposed method aims at reducing migration

overheads while making effective use of cloud resource by combining the optimization-based strategy with pre-copy migration scheme. The following is a summary of this work's main findings:

- To increase the effectiveness of migration in cloud computing settings, an efficient Pre-copy Live Virtual Machine Migration (PLVM) framework is suggested.
- The Fire Hawks Optimization (FHO) algorithm is employed to adaptively determine the optimal Dirty Rate (DR) during the migration process.
- A fitness function is formulated by incorporating important performance indicators to direct the optimization process, such as resource use, downtime, and migration time.
- The proposed approach is implemented using the CloudSimsimulation environment and evaluated using realistic workload traces to represent practical cloud scenarios.
- Results from experiments show that the suggested approach successfully shortens migration times and downtime while improving resource utilization compared with existing migration techniques.

The paper is structured. Section 2 discusses live VM migration, optimization-based migration, and cloud computing. The suggested technique is in Section 3, in addition to the PLVM framework it also provides information about the adaptive Dirty Rate and Fire Hawks Optimization Algorithm Section 4 describes the performance evaluation and experimental setup using CloudSim under NASA workload traces along with results and discussion. To conclude, section 5 summarises this paper and possible future research directions.

2. Related Works

Narander and Swati [16] proposed a new hybrid live VM migration technique this integrates pre-copy and post-copy method elements. Because it does not require waiting for the entire VM image to be transferred before booting the destination VM, this approach is fast. An attached Network-Attached Storage (NAS) reduces overhead between source and destination hosts in the traffic. This is the process used above. Rather than moving the complete memory image of the VM, just the execution log is moved. Furthermore, the formulation of VM clusters is predicated on the similarity exhibited by VM images pertaining to the same application, thus effectively restricting migration distances to within these clusters.

In the pre-copy strategy, an important required activity that minimizes downtime of the service is the selection of an appropriate time period for VM migration. To overcome this challenge, Haris et al. [17] proposed a method for optimizing pre-copy migration based on machine learning. The proposed approach based on triggered random walk and feature selection uses three stages.According to the experimental data, this model efficiently minimizes downtime and service unavailability during migration, outperforming other models in terms of prediction precision.

Yazidi et al. [18] worked on the graph partitioning theory-based VM migration scheduling issue, which groups VMs to ensure high intra-communication between them. After that, an affinity-based scheduling mechanism is established to regulate the order of migration of these groups according to inter-group traffic. This approach reduces the separated traffic volume by more than 30%.

LMEB technique is introduced by Gupta et al. [19] for Live migration with ballooning. It aims at decreasing the amount of data involved in migration resulting in reduced energy use during migration. A simulation study was conducted using specific VM and server configurations, which showed energy (18%), migration time (20%) and downtime (20%) reduction in comparison with existing live migration with ballooning (LMB) approach.

Haris et al. [20] recommended a machine learning-based prediction model for cloud pre-copy live virtual machine migration. The method expects the downtime during migration and stops the stage of iteration when a fixed value is reached by the expected value. Hence, completion of migration improves. The proposed method was implemented and evaluated in a testbed using KVM and QEMU technologies with custom VM and memory-intensive workloads. The findings demonstrate that pre-copy live VM migration performance may be improved by prediction-based optimization.

Table 1: Literature Review Of Cloud Environment Live Vm Migration Techniques

Author & Reference	Method	Merits	Demerits

Kumar Narander and Saxena Swati (2014) [16]	Hybrid Live VM Migration (Pre-copy + Post-copy)	Reduces migration delay by allowing the destination VM to start before full memory transfer. Minimizes network traffic using execution log transfer and VM clustering.	Requires additional NAS infrastructure for storage support. Cluster management increases system complexity in large datacentres.
Raseena M. Haris et al. (2023) [17]	Machine Learning-based Pre-copy Migration Optimization	Improves migration decision accuracy using predictive modeling. Significantly reduces service downtime during VM migration.	Requires large training data and pre-processing effort. ML model training introduces computational overhead.
Anis Yazidi et al. (2018) [18]	Graph Partitioning with Affinity-aware Scheduling	Groups VMs with high communication affinity to reduce network traffic. Improves migration scheduling `efficiency in distributed cloud environments.	Graph partitioning increases computational complexity in large systems. Scheduling overhead may increase migration latency.
Gupta et al. (2022) [19]	Energy-aware Live VM Migration using Ballooning	Reduces energy consumption during migration by decreasing memory transfer size. Improves migration time and system resource utilization.	Memory ballooning may temporarily affect VM performance. Effectiveness depends on available reclaimable memory.

3. Proposed Methodology

This work improves cloud virtual machine migration applications by using pre-copy live virtual machine migration. The proposed method aspires to lessen migration overhead by selecting an optimal Dirty Rate (DR) adaptively during migration. At migration time, memory pages change at Dirty Rate while the VM is operating. An improper choice of DR can greatly increase the number of iterations during migration which increases the migration time and downtime. To rectify this issue, the proposed method determines the optimal dirty rate by utilizing the Fire Hawks Optimization (FHO) algorithm throughout the pre-copy migration process. In this context, the fitness function is defined using three important performance parameters: time, downtime, and resource use during transfer. Optimizing reduces migration time and downtime and maximizes computing resources.Incorporating the optimization algorithm into the pre-copy mechanism, PLVM improves the operational performance and efficiency of VM migration in cloud data centers.

The proposed method's overall workflow is divided into three components: feature extraction, precopy migration modeling, and optimization-based selection of the dirty rate. The next subsections describe these stages in great detail (figure 2).

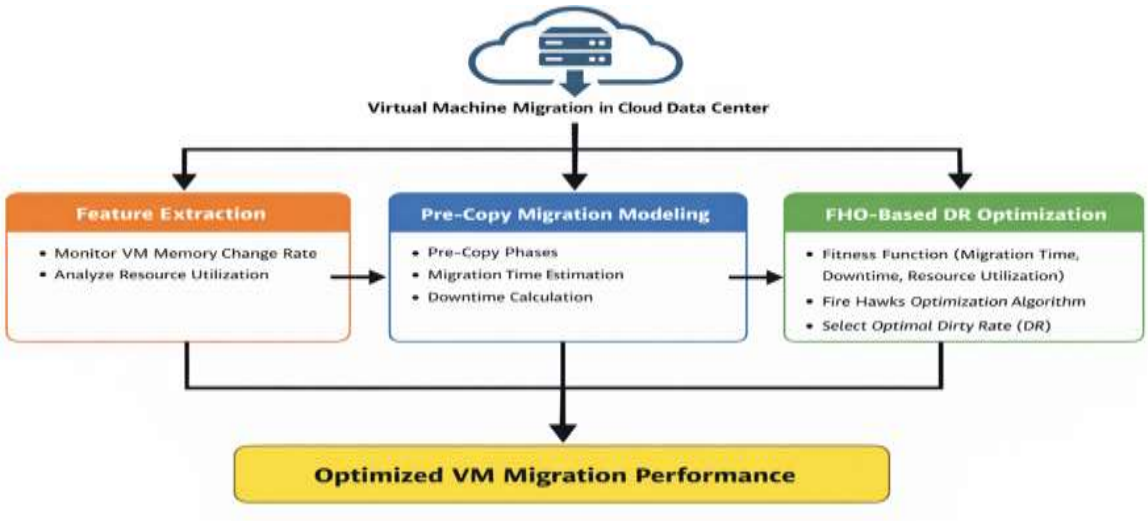

Figure 2: The Overall Process Of The Proposed Methodology

3.1 Feature Extraction

The importance of feature extraction in the migration process of VM has been demonstrated via the identification of key parameters. In the proposed approach, various system and network parameters are

treated as input features for the migration optimization model. The simulation uses many features such as VM size, Dirty Rate (DR), the rate at which a page gets transferred, the CPU utilization by source server, the CPU utilization by destination server, the Memory utilization by source sever, Memory utilization by the destination server, Network utilization of the VM. The parameters reflect workload and resource consumption conditions for the target VM during migration. Among these features, the DR is considered an important parameter because it directly impacts the pre-copy process's amount of their migration iterations.

3.2 Pre-copy Live Migration

Pre-copy is a popular technique for live virtual machine migration in cloud computing. Initially, the destination host receives the VM disk from the source host. At that moment, the VM is still operating on the source computer. While this is taking place, if data is newly written to the disk or memory, then the modified data block is detected and identified as “dirty block” [21]. The modified blocks are transferred once more in the following migration rounds to ensure consistency between the hosts at the source and the destination.

The iteration of moving modified data blocks continues until either the rate of newly generated dirty blocks falls below a maximum number of loops or a threshold value is reached.Additionally, the VM memory state is obtained using a similar repetitive copying after the transfer of virtual-disk contents. At each pass or iteration, only the modified pages in the objective host receives the memory transfer. This iterative methodology aids in restricting the data stream at the ultimate migration phase.

When the remaining modified memory pages set shrinks down to a small enough size, for the last CPU state transfer, source host and remaining memory pages interrupt the VM. Once the final synchronization is done, the VM is started on the destination host. The applications running in the VM continue executing with no interruption. The parameter called dirty rate (DR) determines how often a page on a disk or memory is altered (or unclean). The DR is an important parameter that influences the working of the pre-copy migration scheme.

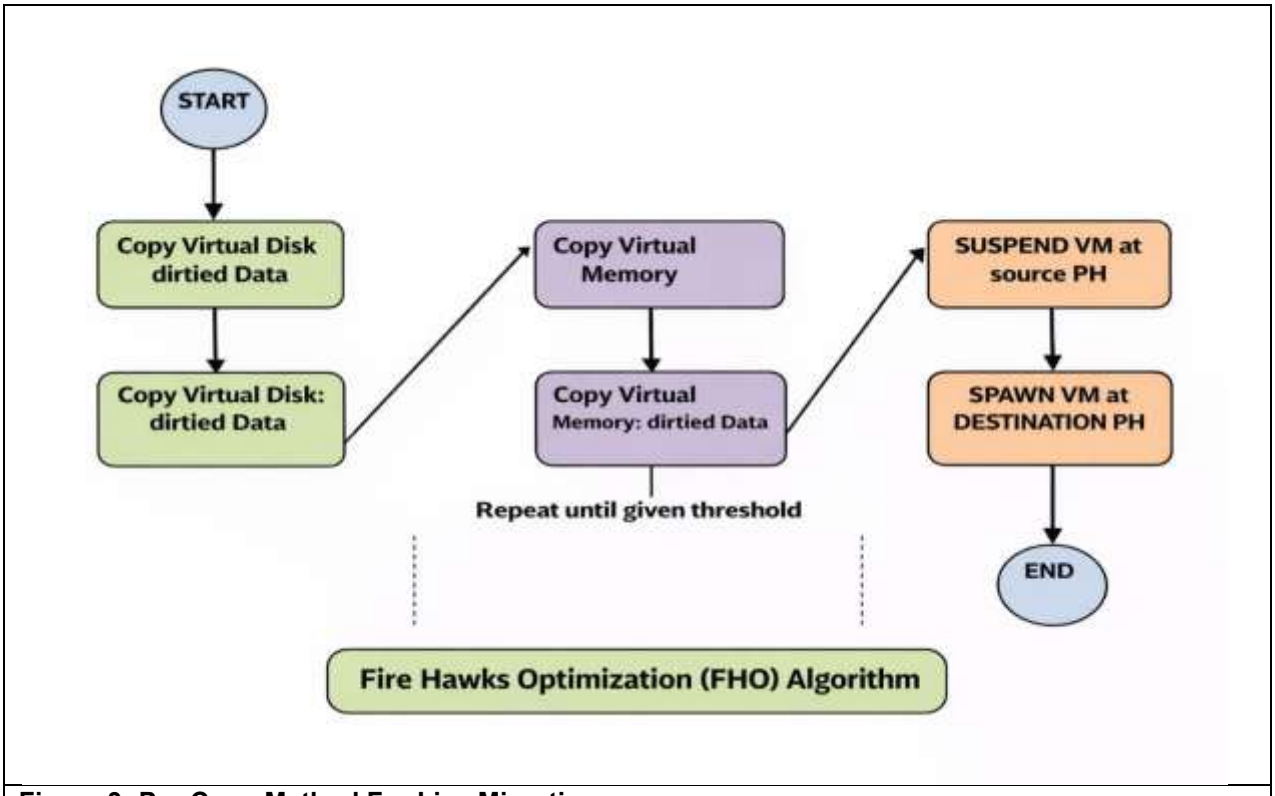

Figure 3: Pre-Copy Method For Live Migration

Figure 3 shows the steps to procedurally perform a pre-copy live migration. As shown in the image, the procedure begins with transferring the virtual disk from the source physical computing system to the destinations machine. The modified disk data transfers. After that, virtual memory and its dirty memory pages visit iteratively until the established migration threshold is reached. In the end, migration process is completed by suspending the VM at the source host and instantiating it at the destination host. Following this sequence,

the service will not be interrupted and the state of the VM will remain consistent on both hosts.(i) Total time taken for migration(M_T)

3.3 Estimation of Parameters

To assess the functioning of the suggested migration plan for virtual machines, many significant parameters are estimated. Such parameters are crucial for analysingthe migration process's efficacy in addition to understanding the system's behavior under different conditions. This study takes into account the three critical parameters of Total migration time, Downtime, and Resource Utilization. These parameters allow to evaluate the efficiency of the migration and effectiveness of system availability and resource usage.

(i) Total Migration Time (MT)

Based on source to destination, total migration time is the time it takes to move the VM. Memory pages are transmitted iteratively during pre-copy migration while the source host's virtual machine operates. This method may modify memory pages many times. They are termed dirty pages. The retrial of these pages must be done in the next iterations until the migration process completes the last step.

Assigned the symbol W to represent the data size to be transferred throughout the migration process measured in (MB). Let R and L denote the total bandwidth available and bandwidth allocated for VM migration, respectively, in (MBps). The value for data transfer time is shown by T which is measured in seconds. Moreover, M is the total RAM size of the virtual machine while x is the amount of RAM transferred during the pre-copy phase for every instance [22].

Equation (1) shows the amount of time it takes to replicate source-to-destination host data. This equation determines the data transfer time based on the allocated migration bandwidth.

$$T_i = w/L \quad (1)$$

The duration required for the initial phase of the pre-copy migration process is computed using Eq. (2).

$$T_{p+z} = \frac{M \cdot \frac{1-(R/L)^N}{1-(R/L)}}{L} \quad (2)$$

After completing the initial transfer, the migration process enters the stop-and-copy phase, where the remaining updated memory pages are transferred by briefly pausing the VM and system state information. The time required for completing this phase is expressed in Eq. (3).

$$T_r = M/L(R/L)^N \quad (3)$$

In Eq. (2) and Eq. (3), the parameter N represents the quantity of pre-copy migration phase iterations. The number of iterations depends on the memory dirty rate and the amount of memory transferred during each cycle. The value of N can be estimated using Eq. (4).

$$N = \min\left(\left[\log_{N/2} \frac{T\perp}{M}, \left\lceil \log_{N/2} \frac{XR}{M(L-R)} \right\rceil\right)\right) \quad (4)$$

By combining the results obtained from Eq. (1), Eq. (2), andEq. (3),it is possible to determine the pre-copy process's total migration time. Thus, Eq. (5) expresses the overall migration time.

$$M_T = T_i + T_{p+z} + T_r \quad (5)$$

This parameter is an important performance indicator because it reflects the migration process's effectiveness and determines how long the migration operation takes to complete in the cloud infrastructure.

(ii) Downtime (DT)

Downtime refers to the unavailability of users for a specific time interval of the VM during the migration process. Even though the pre-copy migration method allows virtual machine operation and copying of portions of memory pages, there still remains a small period when the VM has to be halted to finish migrating the last copy of memory pages [23].

The service interruption is an essential period in the cloud environments as it may affect the availability of the applications and the experience of the users. As a result, an important goal of VM migration strategies minimizes downtime.

The downtime experienced during the migration process is calculated using Eq. (6).

$$D_T = \frac{U_T}{M_T} \quad (6)$$

where U_T represents the time period during which the virtual machine remains unavailable to users, and M_T represents the total migration time. By evaluating this metric, it becomes possible to analyse how efficiently the migration process maintains service continuity.

(iii) Resource Utilization

Another performance evaluation factor for virtual machine migration is how resources are utilized. A cloud data center provides multiple virtual machines some or all of CPU, RAM, and storage are examples of a host's physical resources. Using these resources effectively will enhance the system performance and take on heavy workloads [24].

The resource utilization running on host is calculated using Eq. (7).

$$RU_{i|} = AU_i/CP_i \quad (7)$$

where CP_idenotes the total resource capacity of the host, and AU_i represents the actual VM host resource use. Higher resource utilization indicates efficient use of available infrastructure resources, which contributes to improved system performance during migration.

3.4 Deriving the Fitness Function

A fitness function is formulated through the combinations of performance parameters previously discussed in order to arrive at a suitable dirty rate DR for the pre-copy migration process. This function's primary objective is to minimize migrating time and downtime while achieving maximum utilization of resources.

The fitness function used in this work is defined in Eq. (8).

$$\mathrm{F(DR)} = \mathrm{w_3} \cdot \mathrm{(RU)}/(\mathrm{w_1 M_T} + \mathrm{w_2} \cdot \mathrm{D_T}) \quad (8)$$

In this expression, F(DR) represents the fitness value corresponding to the selected dirty rate. The parameters $\mathrm{M_T}$, $\mathrm{D_T}$, and RU represent the total migration time, downtime, and resource utilization, respectively. The coefficients $\mathbf{w_1}$, $\mathbf{w_2}$, and $\mathbf{w_3}$ are weighting constants that determine the contribution of each parameter in the fitness evaluation. These constants take values in the range (0,1).

The aim of optimization process is to recognise the dirty rate which yields the maximum fitness [25]. A higher fitness value means that the migration process will have less migration time, less downtime and better use of system resources. In order to achieve this optimal design, the proposed framework employs Fire Hawks Optimization (FHO) algorithm, which will be described in succeeding section.

3.5 Fire Hawks Optimization (FHO) Algorithm

The FHO algorithm is a nature-inspired optimization technique that models the hunting and foraging behavior of fire hawks. This algorithm simulates the way fire hawks spread fire in order to flush out prey and make hunting easier. In the optimization context, this behavior is used to investigate and use the search space to determine the most suitable possibility [26]. Initially, a set of candidate solutions is generated, which represent the potential positions in the search area of prey and fire hawks. These candidate solutions are denoted as **A** and are represented as position vectors (Eq 9). The working procedure of the FHO algorithm is illustrated in Figure 4.

The initialization of the position vectors is performed randomly within the defined search space limits. The initialization process is expressed in the following equation (10).

$$A = \begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_k \\ \vdots \\ Z_M \end{bmatrix} = \begin{bmatrix} A_1^1 & A_1^2 & \cdots & A_1^l & \cdots & A_1^d \\ A_2 A_2^2 & \cdots & A_1^l & \cdots & & A_1^d \\ \vdots & \vdots & \vdots & & & \vdots \\ A_k^1 A_k^2 & \cdots & A_1^l & \cdots & & A_1^d \\ \vdots & \vdots & & \vdots & & \vdots \\ Z_M^1 & Z_M^2 & \cdots & Z_M^l & \cdots & Z_M^d \end{bmatrix}, \begin{cases} k = 1,2,\dots, \\ l = 1,2,\dots, \\ l \end{cases} \quad (9)$$

$$a_k^i(0) = a_{k,\min}^i + \mathrm{rand}\left(a_{k,\max}^i - a_{k,\min}^i\right) \begin{cases} k - 1,2,\dots,M \\ l - 1,2,\dots, \\ d \end{cases} \quad (10)$$

where d represents the dimensionality of the search space, and M denotes the total number of candidate solutions. The parameter a_k^i represents the ith parameter of the kth candidate solution, while $a_{k,\max}^i$and $a_{k,\min}^i$show the search space's top and bottom boundaries.

Once the candidate solutions are initialized, each solution is assessed using the fitness function. The candidate solution with the greatest fitness value is the Fire Hawk (FH), with the others being prey. The fire hawks attempt to spread fire strategically to force the prey into positions where they can be easily captured.

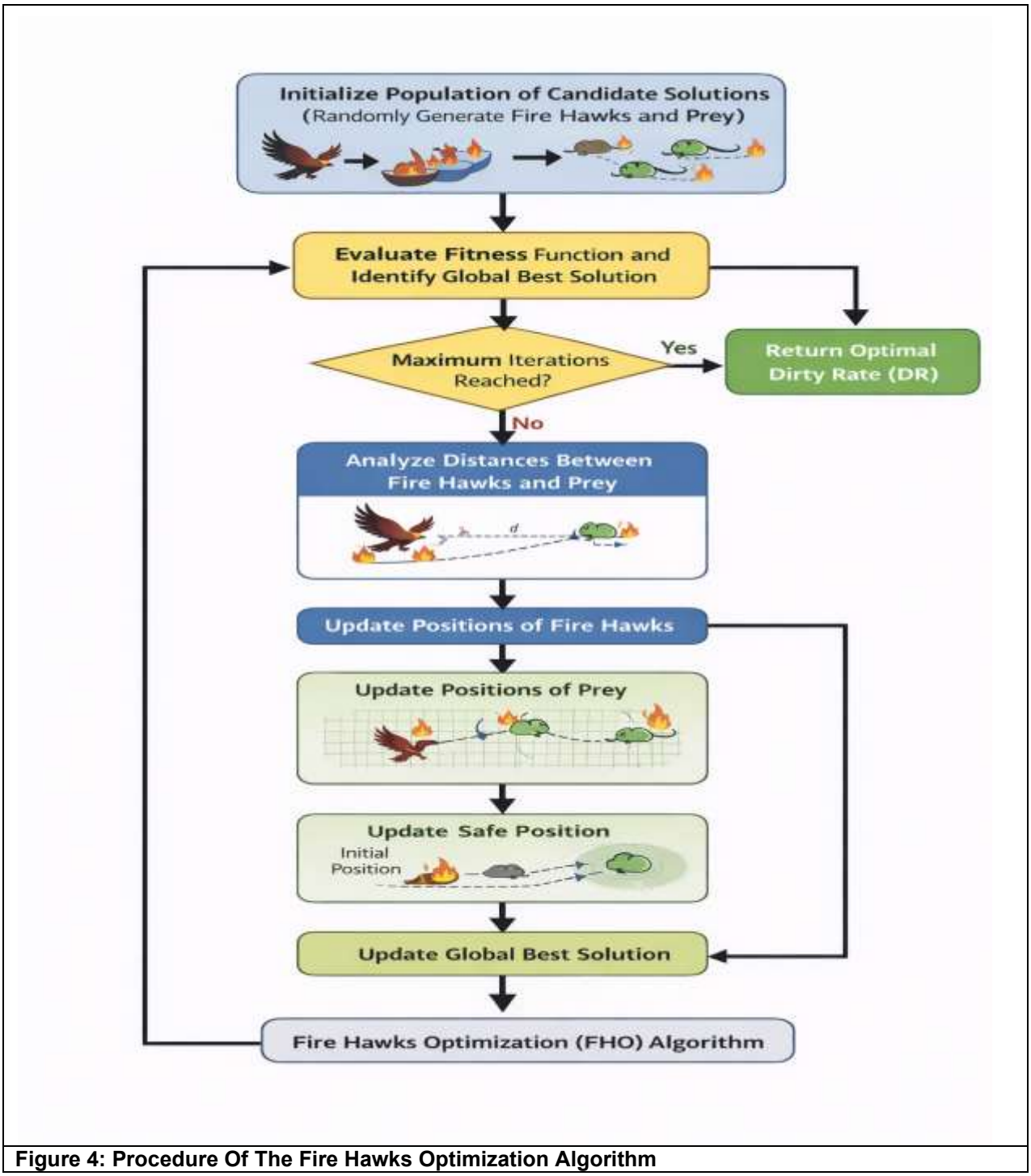

Figure 4: Procedure Of The Fire Hawks Optimization Algorithm

The positions of the prey and fire hawks are represented as in Eq (11):

$$P_r = \begin{bmatrix} P_{r1} \\ P_{r2} \\ \vdots \\ P_{r2} \\ \vdots \\ P_{r2} \end{bmatrix}, 3-1,2 \dots vFH = \begin{bmatrix} FH_1 \\ FH_2 \\ \vdots \\ FH\_ \\ \vdots \\ FH\_ \end{bmatrix}, u = 1,2 \dots m \qquad (11)$$

where P_r represents the ith prey solution, **v** denotes the total number of prey, FH represents the uth fire hawk, and***m*** represents the total number of fire hawks.

Next, fire hawks' closest prey is computed based on distance. This distance calculation helps define the effective hunting territory of each fire hawk. Equation 12 represents the division between the fire hawk and the prey:

$$D_x^0 = \sqrt{(a_2 - a_1)^2 + (b_2 - b_1)^2} \qquad (12)$$

where ($a_1 - a_2$) and ($b_1 - b_2$) show the coordinates of the prey and fire hawk inside the search area.

The FH obtains flaming sticks from the main fire and disperses them over different parts of the search space in the subsequent phase of the optimization process. This behaviour imitates the hunting strategy of fire hawks, where fire is spread out intentionally so as to flush out the prey and make capturing them easier. In the optimization problem, the optimization algorithm is likely to generate variation in the search space at every iteration. Every fire hawk bears a fire stick and puts it in a specific place to direct the motions of surrounding prey solutions. Certain fire hawks seem to retrieve burning sticks from other fire hawks' territories to enhance their hunting ability [27].

The algorithm employs a position updating tactic to imitate the cooperative charateristic of fire hawks. As such, the position updating of a fire hawk depends on the global best solution and the positions of neighbouring fire hawks. This is the mathematical representation of this position updating mechanism in Eq. (13).

$$FH_u^{\text{new}} = FH_a + (r_1 \times G - r_2 \times FH_n) \qquad (13)$$

In Eq. (12),r_1 and r_2 represent randomly generated numbers in the range (0,1), which introduce stochastic behavior into the search process and help maintain diversity among candidate solutions. The term FH_u denotes the current position of the$a$th fire hawk, while FH_a^{new} represents the updated position of that fire hawk after applying the position update rule. The parameter G indicates the global best solution obtained so far during the optimization process, and FH_n represents the nearest fire hawk in the search space. The algorithm finds a compromise between exploration and exploitation by taking into account both the global best solution and the nearby fire hawk positions, which allows it to find the best option rapidly.

Another crucial behavioural feature considered in the Fire Hawks Optimization is the movement of prey solutions in the region controlled by each fire hawk. In nature, when fire spreads across a region, prey animals attempt to escape from the flames and move toward safer locations. However, in some cases, the prey may accidentally move closer to the fire hawks while trying to escape. This unpredictable movement behavior is incorporated into the optimization algorithm as another position updating mechanism for prey solutions**.** The mathematical formulation representing this prey movement is given in Eq. (14)**.**

$$\text{Pr}_{\varphi}^{\ new} = \text{Pr}_{\varphi} + (r_3 \times FH_u - r_4 \times S_n) \qquad (14)$$

In Eq. (14)**,**Pr_{φ}represents the current position of the φth prey solution, while $\text{Pr}_{\varphi}^{\ new}$denotes the updated position of that prey after movement. The parameters $\mathbf{r_3}$ and $\mathbf{r_4}$ are randomly generated numbers that introduce randomness into the prey movement behavior. The variable FH_u represents the position of the uth fire hawk that controls the region in which the prey is located. The term S_p denotes the safety location within the territory of the fire hawk, which represents a potential region where the prey may attempt to escape in order to avoid capture. This update mechanism enables the prey solutions to explore different regions of the search space while reacting to the movement of the fire hawks.

Furthermore, prey may also move across the regions controlled by other fire hawks. In such situations, the prey may attempt to reach safer areas within the search space or accidentally move closer to another fire hawk. This cross-region movement allows the algorithm to increase the diversity of candidate solutions and prevents premature convergence toward local optima. The position updating rule that models this behavior is represented in Eq. (15)**.**

$$\text{Pr}_q = \text{Pr}_q + \left(r_s \times FH_a - r_b \times S_p\right) \qquad (15)$$

In Eq. (15)**,**Pr_q represents the current position of the $_q$th prey solution, while the right-hand side of the equation determines the updated position after considering the influence of other fire hawks. The parameters r_s and r_b are random numbers that help maintain stochastic search behavior. The term FH_a represents the positions of the remaining fire hawks present in the search space, and S_p denotes the safe location toward which the prey may attempt to move.

Finally, the safe locations within the search space are determined based on the average position of prey solutions present in a particular region. These safe locations guide the movement of prey and help maintain in

the optimization process, find a balance between exploration and exploitation. The safe region estimation is expressed using Eq. (16).

$$S_{pi} = \frac{\sum \Pr_r}{r}, S_r = \frac{\sum \Pr_r}{m} \quad (16)$$

In Eq. (16),$\Pr_r$ represents the position of the rth prey solution. The parameters r and m denote the total number of prey and fire hawks, respectively.By determining the average position through the algorithm, the algorithm calculates safe zones in the space. This will help the prey solutions during the later iterations. By repeatedly performing these steps, the Fire Hawks Optimization algorithm refines the candidate solutions and converges towards the optimal value of dirty rate (DR), which maximizes the fitness function that we have defined. The proposed virtual machine migration framework helps achieve lesser migration time, less downtime and better resource utilization due to this optimization process.

Algorithm 1. Fire Hawks Optimization (Fho) For Optimal Dirty Rate Selection

Input:

Population size MMM, number of fire hawks mmm, maximum number of iterations, the upper and lower limitations of the search space are represented by search space limits.

Output:

Optimal Dirty Rate DR∗

Step 1: Initialization

1. Initialize the population of candidate solutions A={a1,a2,…,aM} randomly within the search space limits.
2. Using the specified fitness function (DR), determine each candidate solution's fitness value.
3. Determine which potential solution has the highest global fitness value.

Step 2: Classification of Fire Hawks and Prey

4. Select the top mmm candidate solutions with the highest fitness values and assign them as Fire Hawks (FH).
5. Assign the remaining candidate solutions as Prey (Pr).

Step 3: Iterative Optimization Process

6. For iteration = 1 to MaxIter do
7. For each Fire Hawk FH_a
   a. Identify the nearest fire hawk FHn.
   b. Update the position of the fire hawk using the position update rule based on the global best solution and nearby fire hawk.
   c. Evaluate the updated solution using the fitness function.
8. For each Prey Prq located in the region of fire hawk FHu
   a. Update the prey position according to the movement influenced by the fire hawk and the safe location.
   b. Evaluate the updated prey solution.
9. Allow prey to move toward other fire hawk regions to increase exploration of the search space.
10. Compute the safe location Sr based on the average position of prey solutions in each region.
11. If a better solution is identified, update the Global Best Solution (G).
12. End For

Step 4: Termination

13. Repeat the iterative process until the maximum number of iterations is reached.
14. Return the best solution obtained, which represents the optimal dirty rate DR∗ that maximizes the fitness function while minimizing migration time and downtime.

4. Experimental Results

The proposed FHO based Pre-copy Live VM Migration (FHO-PLVM) technique is implemented in Cloudsim and compared with Machine Learning based Pre-copy Live VM Migration (ML-PLVM) [12] and OMA based Pre-copy Live VM Migration (OMA-PLVM) [13] approaches. The NASA workload was used to replicate the requests made by Web users to the Access Point (AP). The load fluctuations over time are accurately represented by this workload.Every day, the site servers get 100960 user requests. Table 2 shows the experimental parameters employed in this study.

Table 2: Experimental Parameters

Parameter	Value
Work load	NASA traces
Resource Utilization Thresholds	$U^{low-thr} = 20\% and U^{high\_thr} = 80\%$

Response Time Thresholds	$RT^{low-thr} = 200msandRT^{high\_thr} = 1000ms$
Desired Response Time	DRT = 1000ms=1s
Configuration of VMs	Medium and Large
Maximum Limitation on On-demand VM	*Max*VM=10VM
Policy for Task and Resource Scheduling	Time-Shared

4.1 Performance Evaluation

The performance assessment of the suggested FHO-PLVM (Fire Hawks Optimization based Pre-copy Live VM Migration) technique is presented in this part along with the experimental findings. The performance of the proposed method is compared with two existing methods, namely ML-PLVM and OMA-PLVM. The experiments are conducted by varying the number of VMs to be migrated from 10 to 50, and the performance is analyzed using the following metrics:

- Memory Utilization
- Total Downtime
- CPU Utilization
- Total Migration Time

These metrics are widely used in cloud computing environments to evaluate the efficiency of VM migration strategies. The results obtained are presented in tables and graphical plots for visualisation and comparison.

Memory Utilization Analysis

According to Table 3 and Figure 5, memory utilization is obtained for different numbers of migrated virtual machines. Memory utilization is an important measure that shows how much of the system resources are being used during the migration.

Table 3: Results For Memory Utilization

No of VMs	FHO-PLVM	ML-PLVM	OMA-PLVM
10	0.75	0.81	0.84
20	0.72	0.77	0.81
30	0.7	0.75	0.8
40	0.67	0.73	0.78
50	0.65	0.7	0.75

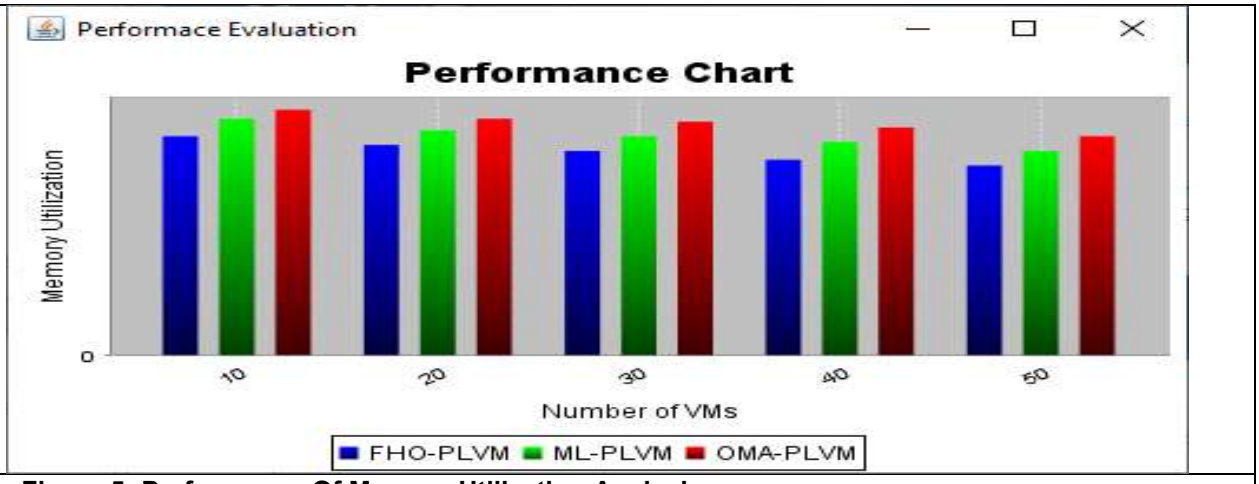

Figure 5: Performance Of Memory Utilization Analysis

Memory utilization increases as the number of VM transferred in the environment also improves as depicted in Figure 5. The aforementioned observation reveals that the proposed FHO-PLVM approach utilizes less memory as compared to ML-PLVM and OMA-PLVM. When there are 50 VMs instead of 10, the memory utilization by all methods goes down gradually. However, the proposed method is the best to offer more optimised memory resource utilization. The average memory utilization of FHO-PLVM is about 7% lower than that of ML-PLVM and 12% lower than that of OMA-PLVM. The improvement suggests that the proposed optimization strategy manages memory effectively during migration, preventing unnecessary waste of resources.

Downtime Analysis

Table 4 and Figure 6 depict the total downtime results for varying numbers of migrated VMs. Downtime refers to the unavailability duration when the virtual machine is not accessible to users during migration.

Table 4: Results For Downtime

No of VMs	FHO-PLVM (ms)	ML-PLVM (ms)	OMA-PLVM (ms)
10	24	33	37
20	22	31	35
30	19	28	32
40	17	26	29
50	16	24	27

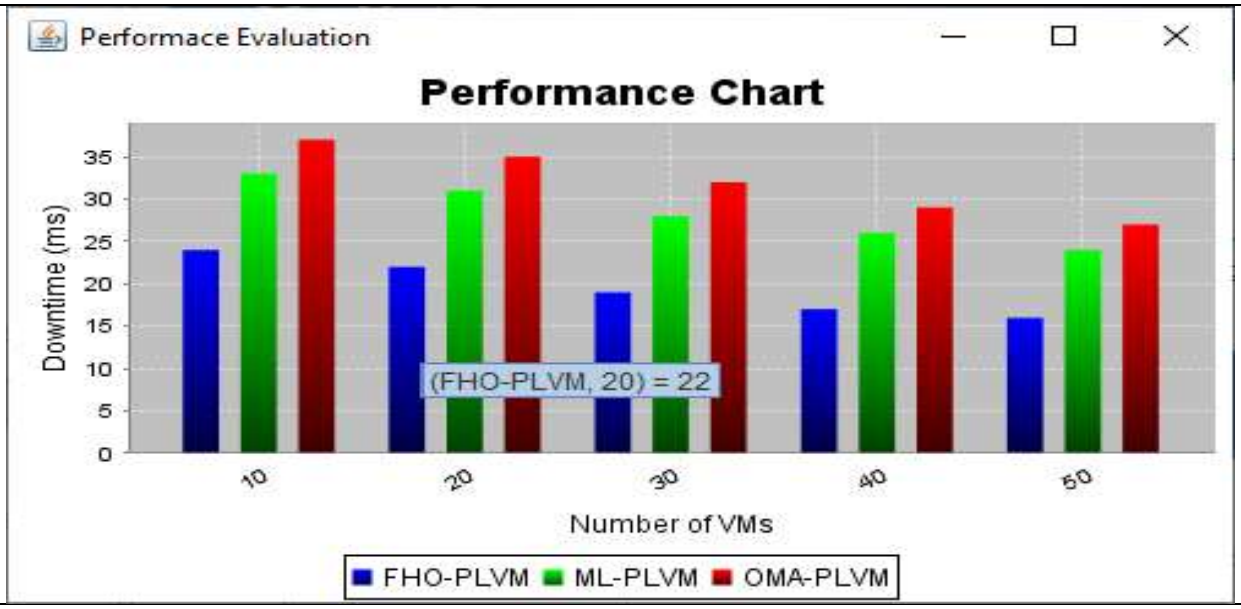

Figure 6: Analysis Of Number Of Vms Vs Total Downtime (Ms)

As we can see in Figure 6, as the number of VM migrations is handled, the downtime also decreases gradually. The proposed FHO-PLVM approach has smaller downtime than existing techniques. The Fire Hawks Optimization algorithm was efficient in selecting dirty rates, thus greatly improving usability of the recommender system. The average downtime of FHO-PLVM is about 31% lower than that of ML-PLVM and 39% lower than OMA-PLVM. When downtime is reduced directly to enhance the availability of a service, a migration process helps in minimizing business disruption to user applications.

CPU Utilization Analysis

The CPU utilization relative to number of virtual machines is depicted in Table 5 and Figure 7. CPU utilization refers to the extent to which the processing resources of the host machines are being used while performing VM migration.

Based on Figure 7, the more VMs there are, the higher the CPU consumption. The CPU utilization attained by the other methods is significantly lower than that of FHO-PLVM. Higher CPU utilization suggests that computational resources are used effectively during the migration. The CPU utilization gained by the proposed scheme is, on average, about 6% higher than ML-PLVM and about 11% higher than OMA-PLVM.

Table 5: Results For Cpu Utilization

No of VMs	FHO-PLVM (%)	ML-PLVM (%)	OMA-PLVM (%)
10	86.55	82.28	75.65
20	88.33	83.73	77.48
30	92.52	85.36	81.98
40	93.11	87.66	83.65
50	95.5	90.11	86.84

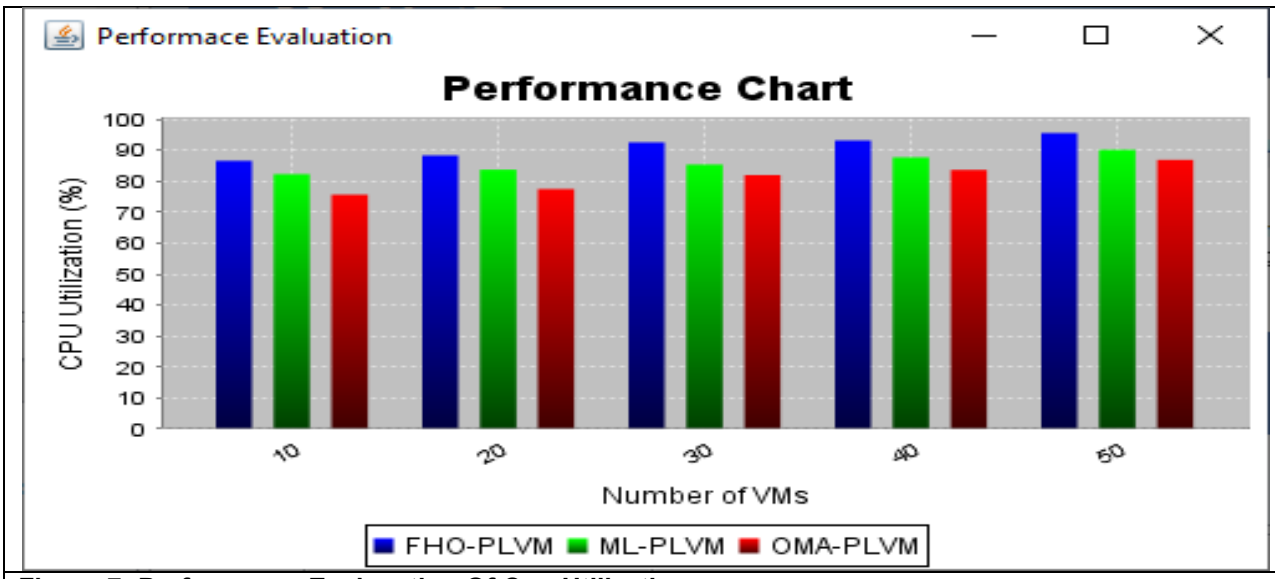

Figure 7: Performance Evaluvation Of Cpu Utilization

Migration Time Analysis

Figure 8 and Table 6 illustrate overall migration time for different VM numbers. Figure 8 demonstrates that the proposed FHO-PLVM when compared to the current techniques, this strategy dramatically shortens the total migration time. When the quantity of VMs increases, the migration time increases for all approaches; however, the proposed method maintains the lowest migration time across all scenarios

Table 6: Results For Migration Time

No of VMs	FHO-PLVM (sec)	ML-PLVM (sec)	OMA-PLVM (sec)
10	1.25	2.58	3.12
20	1.78	2.94	3.45
30	2.25	3.65	4.12
40	2.66	4.15	4.95
50	4.34	5.25	5.31

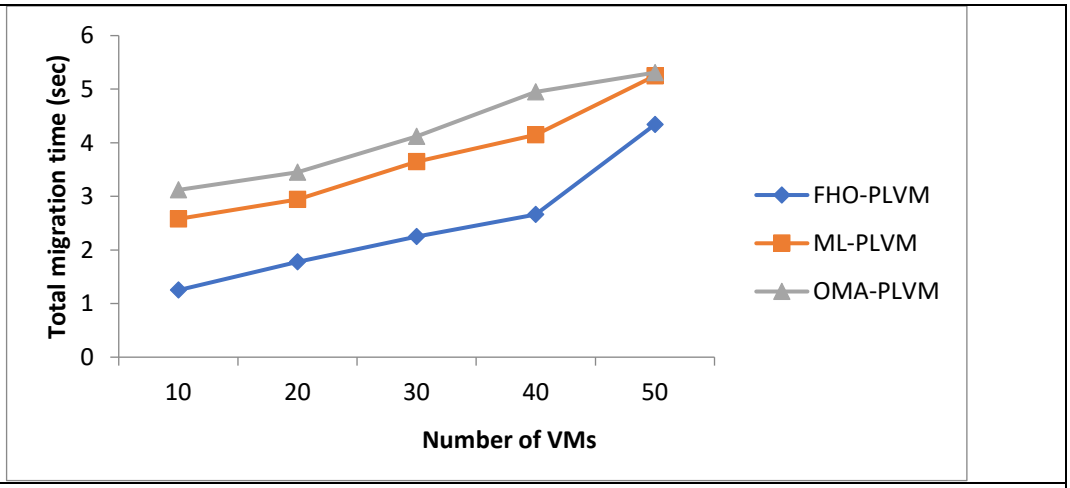

Figure 8: Number Of Vms Vs Total Migration Time (Sec)

This reduction is mainly due to the effective optimization of the dirty rate using the Fire Hawks Optimization algorithm, which minimizes unnecessary memory retransmissions during the pre-copy migration process. On average, the total migration time achieved by FHO-PLVM is approximately 37% lower than ML-PLVM and 44% lower than OMA-PLVM, proving the strategy improves migration efficiency.

5. Conclusion And Future Work

In this paper, a Fire Hawks Optimization based Pre-copy Live Virtual Machine Migration (FHO-PLVM) technique has been proposed to increase cloud computing VM migration.During pre-copy migration, the Fire Hawks Optimization (FHO) algorithm adaptively determines the best filthy rate. The optimization method includes migration time, downtime, memory use, and CPU usage.By choosing appropriate dirty rates, the proposal limits unnecessary memory retransmission at the time of migration. The effectiveness of the proposed model FHO-PLVM was checked using CloudSim Simulation. We utilized existing ML-PLVM and OMA-PLVM techniques to compare the performance of the proposed method. The trial demonstrated that the suggested strategy shortens downtime and migration times. The memory and CPU utilization is improved during model migration as well. According to these findings, FHO-PLVM is an effective solution for live migration of virtual machines in cloud infrastructures. The suggested approach might be enhanced in the future of cloud computing by using advanced machine learning methods to predict the workload dynamically to enhance migration efficiency at large scale cloud data centers.

References

1. Ruprecht, A., Jones, D., Shiraev, D., Harmon, G., Spivak, M., Krebs, M., Baker-Harvey, M. and Sanderson, T., 2018. VM live migration at scale. *ACM SIGPLAN Notices*, *53*(3), pp.45-56.
2. Surbiryala, J. and Rong, C., 2019, August. Cloud computing: History and overview. In *2019 IEEE cloud summit* (pp. 1-7). IEEE.
3. He, T., Toosi, A.N. and Buyya, R., 2019. Performance evaluation of live virtual machine migration in SDN-enabled cloud data centers. *Journal of Parallel and Distributed Computing*, *131*, pp.55-68.
4. Basu, D., Wang, X., Hong, Y., Chen, H. and Bressan, S., 2019. Learn-as-you-go with megh: Efficient live migration of virtual machines. *IEEE Transactions on Parallel and Distributed Systems*, *30*(8), pp.1786-1801.
5. Fernando, D., Terner, J., Gopalan, K. and Yang, P., 2019, April. Live migration ate my VM: Recovering a virtual machine after failure of post-copy live migration. In *IEEE INFOCOM 2019-IEEE conference on computer communications* (pp. 343-351). IEEE.
6. Addya, S.K., Turuk, A.K., Satpathy, A., Sahoo, B. and Sarkar, M., 2018. A strategy for live migration of virtual machines in a cloud federation. *IEEE Systems Journal*, *13*(3), pp.2877-2887.
7. Alyas, T., Javed, I., Namoun, A., Tufail, A., Alshmrany, S. and Tabassum, N., 2022. Live Migration of Virtual Machines Using a Mamdani Fuzzy Inference System. *Computers, Materials & Continua*, *71*(2).
8. Li, C., Feng, D., Hua, Y. and Qin, L., 2019. Efficient live virtual machine migration for memory write-intensive workloads. *Future Generation Computer Systems*, *95*, pp.126-139.
9. Kirov, D.E., Toutova, N.V., Vorozhtsov, A.S. and Andreev, I.A., 2021. Feature selection for predicting live migration characteristics of virtual machines. *T-Comm-Телекоммуникации и Транспорт*, *15*(7), pp.62-70.
10. Tan, D., Liu, F., Xiao, N. and Xie, Y., 2019, May. Optimizing virtual machine live migration in distributed edge servers based on hybrid memory. In *2019 International Conference on High Performance Big Data and Intelligent Systems (HPBD&IS)* (pp. 29-34). IEEE.
11. Devi, N.N. and Kumar, S.V., 2023. Pre-copy live vm migration techniques in cloud computing using hdwhm algorithm. *International Journal on Information Technologies & Security*, *15*(1).
12. Jalaei, N. and Safi-Esfahani, F., 2021. VCSP: virtual CPU scheduling for post-copy live migration of virtual machines. *International Journal of Information Technology*, *13*(1), pp.239-250.
13. Chen, Y., Xu, S., Yang, H., Zhou, R., Guo, D. and Zhou, Q., 2024. Live migration of virtual machines based on dirty page similarity. *IEEE Transactions on Cloud Computing*, *12*(2), pp.563-579.
14. Jain, P., 2021. *Optimized pre-copy live virtual machine migration for memory-intensive workloads* (Doctoral dissertation, Dublin, National College of Ireland).
15. Chanchio, K. and Yaothanee, J., 2018, April. Efficient pre-copy live migration of virtual machines for high performance computing in cloud computing environments. In *2018 3rd International Conference on Computer and Communication Systems (ICCCS)* (pp. 497-501). IEEE.
16. Narander, K. and Swati, S., 2014. An Efficient Live VM Migration Technique in Clustered Datacenters. *Research Journal of Recent Sciences ISSN*, *2277*, p.2502.
17. Haris, R.M., Khan, K.M., Nhlabatsi, A. and Barhamgi, M., 2024. A machine learning-based optimization approach for pre-copy live virtual machine migration. *Cluster Computing*, *27*(2), pp.1293-1312.
18. Yazidi, A., Ung, F., Haugerud, H. and Begnum, K., 2018. Effective live migration of virtual machines using partitioning and affinity aware-scheduling. *Computers & Electrical Engineering*, *69*, pp.240-255.
19. Gupta, N., Gupta, K., Qahtani, A.M., Gupta, D., Alharithi, F.S., Singh, A. and Goyal, N., 2022. Energy-aware live VM migration using ballooning in cloud data center. *Electronics*, *11*(23), p.3932.
20. Haris, R.M., Barhamgi, M., Nhlabatsi, A. and Khan, K.M., 2024. Optimizing pre-copy live virtual machine migration in cloud computing using machine learning-based prediction model.

21. Xu, D., Li, L., Zhang, H., Li, J. and Xu, H., 2022. An Optimized Dynamic Migration Method of Virtual Machine. *Journal of Circuits, Systems and Computers*, *31*(06), p.2250101.
22. Elsaid, M.E., Abbas, H.M. and Meinel, C., 2020. Live Migration Timing Optimization for VMware Environments using Machine Learning Techniques. In *CLOSER* (pp. 91-102).
23. Bhardwaj, A. and Rama Krishna, C., 2018, September. Improving the performance of pre-copy virtual machine migration technique. In *Proceedings of 2nd International Conference on Communication, Computing and Networking: ICCCN 2018, NITTTR Chandigarh, India* (pp. 1021-1032). Singapore: Springer Singapore.
24. Chen, J., Teng, Y., Ma, S., Zhong, T. and Yi, M., 2024, September. Enhancing Pre-Copy Strategy for Efficient Containerized Stateful Service Migration in MEC. In *2024 IEEE 35th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)* (pp. 1-6). IEEE.
25. Azizi, M., Talatahari, S. and Gandomi, A.H., 2023. Fire Hawk Optimizer: A novel metaheuristic algorithm. *Artificial Intelligence Review*, *56*(1), pp.287-363.
26. HS, M., T, S.K., Gupta, P. and McArdle, G., 2023. A Harris Hawk Optimisation system for energy and resource efficient virtual machine placement in cloud data centers. *Plos one*, *18*(8), p.e0289156.
27. Elsaid, M.E., Abbas, H.M. and Meinel, C., 2020. Live Migration Timing Optimization for VMware Environments using Machine Learning Techniques. In *CLOSER* (pp. 91-102).
28. Moti Ranjan Tandi. (2026). Stochastic Wind Field Learning Using Multi-Fidelity Surrogate Models for Robust Micro-Siting Optimization. *Journal of Scalable Data Engineering and Intelligent Computing*, 1-8.
29. F Rahman. (2026). Scalable Safety-Constrained Learning Pipelines for Distributed Digital-Twin-Based Energy Optimization in Large-Scale Electric Mobility Systems. *SECITS Journal of Scalable Distributed Computing and Pipeline Automation*, 1-8.
30. Nidhi Mishra. (2026). Fault-Tolerant Learning-Assisted Predictive Control Protocols for Real-Time Trajectory Coordination. *Transactions on Secure Communication Networks and Protocol Engineering*, 1-7.
31. T.G. Zengeni, M.P. Bates. (2026). Fractional Nonlinear Dynamics of Heat and Mass Transfer in Porous Industrial Reactors. *Applied Nonlinearity in Science and Technology*, 7-12.