Traffic-Aware Live Virtual Machine Migration

Interim Report

SCS 4224: Final Year Project in Computer Science

Venudi Dayaratne

20000286

November 11, 2024

Declaration

This report is my original work and has not been submitted any university or institute. To the best of my knowledge, it does not include any material authored or published by others, except where appropriately cited in the text.

Name : H.B.V.D. Dayaratne
Registration No. : 2020/CS/028
Index No. : 20000286

[illegible]

Signature

Supervisor : Dr. (Mrs). D.K. Fernando

Fernando

Signature

Contents

List of Figures

Abbreviations

CDC Cloud Data Center. 7, 9

DT Downtime. 5

LAN Local Area Network. 18

NFS Network File System. 18

NIC Network Interface Card. 9, 14, 17

NTVMM Network Traffic-based Virtual Machine Migration algorithm. 10

OS Operating System. 1, 14

QoS Quality of Service. ii, 4, 9

SLA Service Level Agreement. 3, 4, 16, 23, 25, 27

SLAs Service Level Agreements. ii, 4, 9

TMT	Total Migration Time. 5
VM	Virtual Machine. iv, 1–6, 9–12, 14, 17, 19, 21, 22, 24, 25, 28
VMs	Virtual Machines. 1, 3–5, 7, 9–11, 14, 15, 17, 19, 23–25, 27, 29
WAN	Wide Area Network. 1
WWS	Writable Working Set. 2, 5

1 Introduction

In today's digital landscape, the need for on-demand and scalable computing resources has driven the adoption of cloud computing. Virtualization enables the creation of virtual images of physical resources, allowing multiple machines to access them simultaneously. A Virtual Machine (VM) emulates a physical machine with its own Operating System (OS), CPU, memory, and functions. A physical machine, known as a node in virtualization, serves as a server hosting one or many VMs. These VMs can be resized and relocated—a process known as VM migration (Bahrami, Haghighat, and Gholipoor n.d.).

Migrating a VM instantaneously while it is running is called *Live Migration*. Migration of multiple VMs can be done simultaneously, which is referred to as *Gang Live Migration*, or sequentially. This process typically involves transferring the VM's memory, disk storage, and network connections from the source node to the destination (Le 2020). Within a Local Area Network (LAN), VM migrations do not require the migration of disk storage due to the use of Network Attached Storage (NAS) (Jo et al. 2013). However, migrations over a Wide Area Network (WAN) necessitate the transfer of both the disk state and memory (Wood, Ramakrishnan, et al. 2014).

There are three (3) traditional live migration techniques namely pre-copy (Clark et al. 2005; Nelson, Lim, Hutchins, et al. 2005), post-copy (Hines and Gopalan 2009; Hines, Deshpande, and Gopalan n.d.), and hybrid-copy (Sahni and Varma 2012). These techniques are designed for single VM migrations. However, when adapting these techniques for multiple VM migrations, if all the VMs monotonously adopt a single migration algorithm, it overlooks the traffic contention. The traffic generated due to the VM migration adds to the existing application traffic sharing the same bandwidth leading to inefficiencies.

1.1 Background

1.1.1 Live VM Migration

In live migration, the memory pages which belong to the VM and disk block are copied. The VM's runtime state (CPU, disk I/O, memory state) should be moved into the destination host node while the VM is still running (Clark et al. 2005).

1.1.1.1 Pre-copy Migration Technique

In the pre-copy live VM migration technique, while the VM operates on the source, its memory pages are moved to the destination. The measurement taken to pause the *memory transfer phase* of the migration is identifying a Writable Working Set (WWS) (Hines, Deshpande, and Gopalan n.d.). Once this milestone is reached this phase stops and thus begins the next phase of the pre-copy migration, the stop-and-copy phase. The VM is temporarily halted facilitating the CPU state and the dirty memory pages to be transferred to the target host. Finally, all connections in the open network are reconfigured to the new host.

1.1.1.2 Post-copy Migration Technique

Post-copy migration copies the non-pageable memory, network states, and minimal CPU to the destination host and resumes the VM at the destination. Following this, the remaining pages are fetched (Bahrami, Haghighat, and Gholipoor n.d.).

1.1.1.3 Hybrid-copy Migration Technique

To mitigate the performance degradation associated with the post-copy approach and the downtime characteristic of the pre-copy method, the hybrid-copy migration technique was developed. This method integrates both pre-copy and post-copy approaches. Initially, while the VM operates on the source host, its memory is copied to the destination as in the first round of the pre-copy technique. Subsequently, the VM is transferred to the destination host and resumes execution. Any page faults occurring thereafter are handled by requesting the dirty memory pages from the source, similar to the post-copy technique (Sahni and Varma 2012).

1.1.2 Multiple VM Migration

Virtual machines residing on the same physical host are known as co-located VMs. These VMs communicate and complement each other's services and executing on the same physical machine reduces communication costs (Huang et al. 2007) and improves consolidation (Wood, Shenoy, et al. 2007). For reasons such as saving energy and maintaining system performance, the need for migrating multiple such correlated VMs (Sun, Liao, Zhao, et al. 2015) arises. These multiple VM migrations can be carried out either serially or parallelly (Callegati and Cerroni 2013; Chang, Walters, and Wills 2013; Sun, Liao, Anand, et al. 2016).

In the *serial migration* strategy, VMs are migrated one after the other using the migration approaches proposed for single VM migration. The live migration of multiple VMs parallelly is called *gang migration* (Deshpande, Wang, and Gopalan 2011). As VMs are migrated concurrently in this approach, the available bandwidth is shared among the VMs.

In gang migration, the total migration time is longer than in serial migration, but the downtime is shorter. This suggests a tradeoff: parallel migration has the advantage of less downtime, making it better for maintaining service availability. However, serial migration has a shorter overall migration time, making it more efficient in terms of communication resources and transmission overhead (Callegati and Cerroni 2013)

Priority level refers to how quickly VMs need to be migrated. Highly prioritized migrations may require gang migration, which, despite its impact on performance, minimizes downtime. For less prioritized migrations, a serial approach might be more suitable, providing a more controlled and gradual process. System administrators can specify the priority level for each migration (Fernando, P. Yang, and H. Lu 2020) or priority can be assigned considering the Service Level Agreement (SLA) violations and migration time violations (Tsakalozos et al. 2017).

Different migrations have varying time requirements and corresponding deadlines, especially when dealing with multiple VMs (Nadeem, Elazhary, and Fadel 2018). Some virtual network functions (VNFs) require rapid migration to maintain low end-to-end latency, while web services with higher latency tolerance can be scheduled over a longer period. For example, in case of a breakdown, the VM migration needs to occur as quickly as possible in contrast to routine server maintenance which isn't as urgent but needs to be done with the least service disruption. Service Level Agreements (SLAs) and Quality of Service (QoS) considerations are crucial in determining whether to use serial or parallel migration for multiple VMs and in determining the bandwidth to be reserved for the migration. Depending on the use case, migrations may need to meet specific deadlines or involve trade-offs between priority level and performance.

1.1.3 Service Level Agreements (SLAs) and Quality of Service (QoS)

Service Level Agreements (SLAs) are formal contracts established between cloud service providers and their customers, defining the expected levels of service delivery, including performance, availability, and redundancy standards (K. Lu et al. 2013). These agreements outline the obligations of both parties and describe the penalties for any failure to meet the agreed-upon terms (Odun-Ayo, Udemezue, and Kilanko 2019). Adherence to metrics defined by SLAs such as response time, availability, and reliability is critical (Gao et al. 2014). Providers must avoid SLA violations and ensure customer satisfaction while reducing resource usage and service interruptions.

Quality of Service (QoS) represents the measurable performance characteristics guaranteed by service providers to meet customer expectations (K. Lu et al. 2013). By ensuring consistent QoS levels, service providers can meet their SLA obligations while optimizing network and resource utilization during the migration process.

1.1.4 Performance Metrics

Performance metrics serve as an essential evaluation to assess the live VM migration methods. Among the numerous metrics identified, total migration time, downtime, total migrated data, migration overhead, application degradation, preparation time, and resume time certain metrics stand out as most commonly used (Voorsluys et al. 2009; Altahat et al. 2020; Soni and Kalra 2013; Hines and Gopalan 2009; Kuno, Nii, and Yamaguchi 2011). Specifically,

- Total Migration Time (TMT) - the time required to complete the entire migration process. In gang migration, it's the time between the beginning of the migration process at the source to the completion of the last VM's migration at the destination.

- Downtime (DT) - the duration during which the services provided by the VMs become inaccessible to users. The VM's CPU state and the WWS are transferred during this time and therefore the CPU execution is fully suspended.

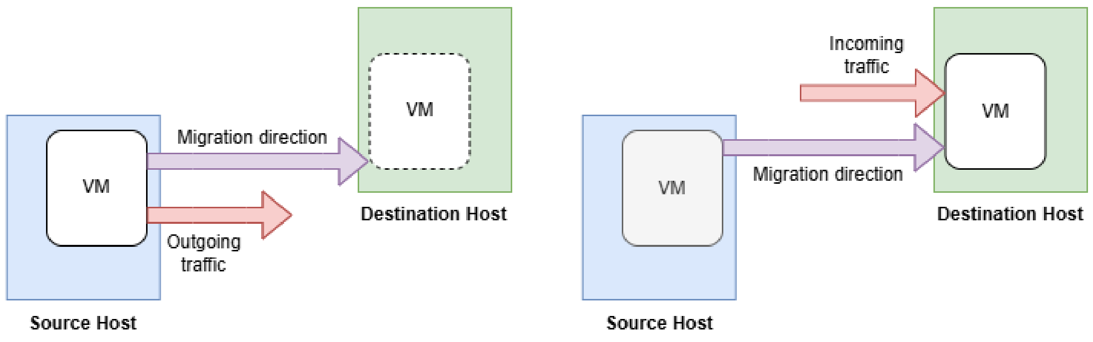
- Application performance degradation - the slowdown of the applications running on the migrating VMs during the migration.

1.1.5 Traffic Contention Problem

In the context of migration, traffic can be classified between workload traffic and migration traffic. As for workload traffic, it includes workloads of the migrating VM, other co-located VMs or business traffic generated by applications that reside in the VM or the host.

1.1.5.1 Migration traffic

- Pre-copy Migration iteratively transfers the state data (CPU, memory, and I/O state) of a VM from the source host to the destination host while the VM continues to run on the source. The traffic is primarily outbound, as data flows from the source to the destination (shown in Figure 1a).



(a) Pre-Copy (b) Post-Copy

Figure 1: Traffic directions in Pre-Copy and Post-Copy migrations

- In Post-copy Migration, the VM is immediately moved to the destination host, and any missing data pages are fetched from the source as needed. The traffic is primarily inbound, with data being requested from the source to the destination (shown in Figure 1b).

1.1.5.2 Application Traffic

- **Incoming traffic** refers to the data packets received by a system or network interface. In VMs and applications, incoming traffic typically includes requests, data, or communications initiated by external entities (Cui, Zhu, et al. 2020). This traffic is critical for the operation of client-server applications and services hosted within the VMs, as it represents the demand side of the service interactions.

- **Outgoing traffic** refers to the data packets transmitted from a system or network interface. For VMs and applications, outgoing traffic includes responses, data, or communications sent out from the VMs to external entities. Outgoing traffic is essential for delivering services and content from the VMs to the end users, forming the supply side of the service interactions (Deshpande and Keahey 2017).

1.1.5.3 Traffic Contention Problem

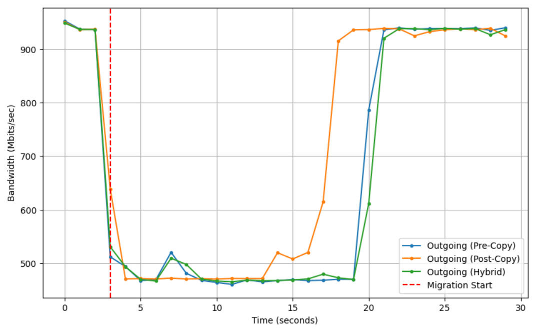
Both migration traffic and workload traffic share the same network bandwidth (shown in Figure 1). When they occur simultaneously, they compete for the limited network resources (Deshpande and Keahey 2017). In pre-copy migration, the outbound migration traffic competes with the outbound application traffic originating from VMs at the source node (shown in Figure 2a), which can degrade both the migration process and the application performance (shown in Figure 3). In post-copy migration, the inbound migration traffic at the destination competes with the inbound application traffic to VMs (shown in Figure 2d, leading to similar degradation in performance and increased migration time (shown in Figures 3) (Cui, Zhu, et al. 2020).

Prolonged migration duration due to network contention can delay the completion of the migration process, thereby delaying the release of resources at the source host. In gang migration scenarios, where multiple VMs are migrated concurrently, the combined migration traffic can severely impact the available bandwidth, further worsening the contention problem.

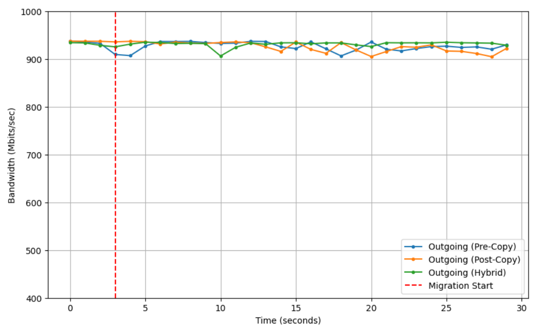
Network-bound applications experience degraded performance due to the reduced bandwidth available for their traffic, leading to slower response times and potentially affecting the quality of service. Both client-server applications and other network-intensive applications running on the same host can suffer from reduced throughput and increased latency due to this directional traffic contention.

1.2 Motivation

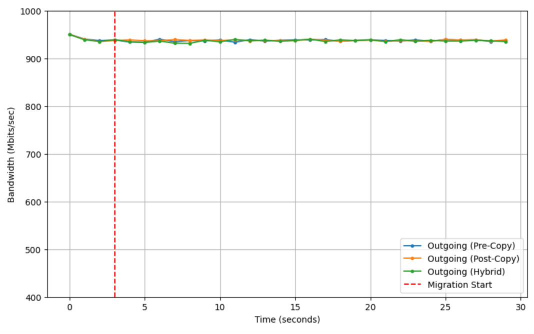
Live migration of VMs within a Cloud Data Center (CDC) can be triggered by several factors, such as balancing server load (Padala et al. 2007; Wood, Shenoy, et al. 2007), managing hardware failures (Nagarajan et al. 2007), performing essential hardware maintenance (Devi, Aruna, Priya, et al. 2011), or achieving energy and power savings through workload consolidation (Nathuji and Schwan 2007; Hu et al. 2008). In each of these scenarios, it is critical that migrations are executed with minimal delay.



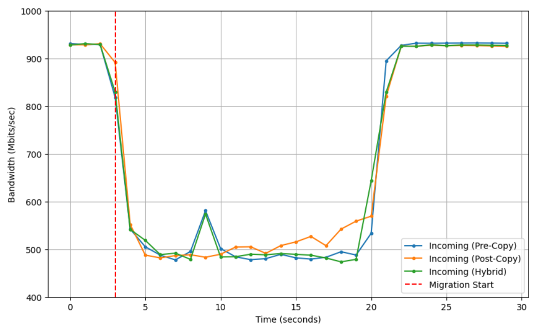
(a) Outgoing traffic at the source node



(b) Incoming traffic at the source node



(c) Outgoing traffic at the destination node



(d) Incoming traffic at the destination node

Figure 2: Bandwidth fluctuations of live migration with incoming and outgoing traffic

When VM migration occurs over the same Network Interface Card (NIC) interface that handles the VM's workload, both types of traffic share bandwidth. This leads to network contention, which is a significant obstacle in live migration—particularly when using pre-copy and post-copy approaches. In pre-copy migration, the outgoing network traffic of the source competes with the migration traffic, whereas in post-copy migration, the incoming network traffic at the destination contends with the migration stream (Deshpande and Keahey 2017). This competition can degrade the performance of applications running on the migrating VM, slow down the migration process, and increase downtime. The rising volume of network traffic in CDCs due to continuous growth and evolution has only exacerbated this issue, further hindering migration efficiency.

Moreover, migrations often come with diverse time constraints and corresponding deadlines, particularly when multiple VMs are involved (Nadeem, Elazhary, and Fadel 2018). Service Level Agreements (SLAs) and Quality of Service (QoS) considerations play a key role in determining these requirements. Depending on the use case, migrations must either meet strict deadlines or balance trade-offs between priority levels and performance outcomes.

Addressing these challenges effectively in live VM migration requires a careful consideration of both network traffic and migration priority.

2 Background

2.1 Literature Review

2.1.1 Traffic-Aware Migration

Shrivastava et al. 2011 introduced AppAware, a live migration approach that optimizes migration destinations by considering network topology, communication patterns of VMs, and physical server limitations, focusing on application-specific communication needs. Similarly, Cui, Z. Yang, et al. 2017 proposed a topology-adaptive Data Center Network (DCN) that constructs and adjusts network topologies dynamically based on VM demands and traffic patterns. Tso et al. 2014 developed SCORE, an approach designed to minimize overall inter-VM communication by treating communication as an optimization problem, using a distributed migration technique that adapts to traffic fluctuations. Collectively, these approaches aim to optimize network topology and communication patterns to reduce network traffic during migration.

Other strategies extend traffic-awareness to load balancing and resource monitoring. For example, Kanniga Devi, Murugaboopathi, and Muthukannan 2018 proposed the System and Traffic-Aware Live VM Migration for Load Balancing (ST-LVM-LB), a graph-based approach that monitors resources and balances load by migrating the least-loaded VMs while accounting for network bandwidth and active traffic flows. Fu et al. 2019 introduced NTVMM, which reduces traffic and network load through algorithms that prioritize high-traffic-generating VMs during selection and placement. Addressing network congestion, Liu et al. 2015 formulated cross-site live migration of multiple VMs as a Mixed-Integer Linear Programming (MILP) problem, factoring in migration traffic and inter-VM communication, and developed the *MinUti-O* algorithm to mitigate complexity. Furthermore, Nasim and Kassler 2015 proposed using multipath-TCP and queue management strategies to enhance traffic-aware live migration, reducing both downtime and migration duration. These approaches emphasize load balancing, resource management, and congestion mitigation through various optimization methods.

Additional traffic-aware solutions have targeted container and VM migration optimization. Maheshwari et al. 2018 introduced ShareOn, a traffic-aware container migration algorithm that selects target nodes based on real-time traffic conditions to minimize network impact. Deshpande and Keahey 2017 proposed a traffic-aware VM live migration algorithm that monitors both application and migration traffic to determine the optimal use of pre-copy or post-copy techniques. Cui, Zhu, et al. 2020 developed an adaptive migration algorithm selection framework using fuzzy clustering to categorize VMs by business traffic, enabling dynamic selection of the most suitable migration approach. These methodologies adapt migration strategies based on current traffic conditions, optimizing migration performance through adaptive mechanisms.

2.1.2 Priority-Aware Migration

There exists research that focus on migration priority as well. Fernando, P. Yang, and H. Lu 2020 introduced a live migration approach that reserves bandwidth according to the urgency of migration. Nadeem, Elazhary, and Fadel 2018 proposed a prioritization-based approach where only low-priority tasks are selected for migration, retaining high-priority ones at the host. Dalvandi, Gurusamy, and Chua 2015 presented an algorithm that considers the requested migration time of VMs for optimal placement and routing. Haidri et al. 2022 explored migration strategies that consider request deadlines to balance load and meet specific timing requirements, while Son and Buyya 2018 focused on VM/flow placement based on migration deadlines.

2.1.3 Traffic Shaping

The use of traffic shaping to optimize VM migrations has been well-documented. The mVM Scheduler, for instance, uses a network model that continuously monitors traffic patterns and network capacity to determine optimal migration start times and bandwidth allocation, dynamically adjusting between parallel and serial migrations based on real-time conditions (Kherbache, Madelaine, and Hermenier 2017). Similarly, Software-Defined Networking (SDN) controllers offer real-time traffic flow management, bandwidth allocation, and route adjustments to prevent link congestion. Network Function Virtualization (NFV) further enhances traffic optimization by prioritizing critical migration flows with virtualized middleboxes (Manzalini et al. 2013).

The Geometric Programming Model offers another perspective by adapting bandwidth to different phases of migration, using a cost function to balance downtime and resource usage, ultimately minimizing network strain and migration time. This model's rapid convergence to optimal values makes it suitable for real-time, multiple VM migration scenarios (Cerroni and Esposito 2016). Pre-copy migration techniques, which progressively increase transfer rates across rounds to minimize residual data, exemplify adaptive bandwidth adjustments that enhance migration efficiency (Choudhary et al. 2017).

Ongoing research in traffic-sensitive live VM migration emphasizes a range of approaches, from selecting migration techniques based on traffic patterns to reducing overall traffic and optimizing migration sequences. Traffic-aware solutions, adaptive topologies, and urgency-based strategies highlight advancements in enhancing migration efficiency. Despite these approaches, integrating traffic and urgency considerations to maximize network performance remains an area for exploration and further improvement.

2.2 Research Gap

Currently, there are certain algorithms that consider traffic contention and some that focus on urgency. However, there is a need for algorithms that take both factors into account. Such an approach would dynamically choose migration strategies based on traffic contention (using pre-copy or post-copy techniques) and priority. This would accommodate tasks that require minimal migration time, those that prioritize reduced downtime, and those willing to trade off between migration time and downtime for better application performance.

Furthermore, comprehensive studies on the decision-making process for performing serial versus parallel migrations are lacking. This research aims to fill that gap by developing algorithms that consider both network traffic and urgency to optimize migration strategies, thereby reducing migration time, downtime, and application performance degradation.

2.3 Research Questions

1. What is the optimal migration strategy to reduce the migration time and minimize network contention, considering how quickly the VM needs to be migrated?

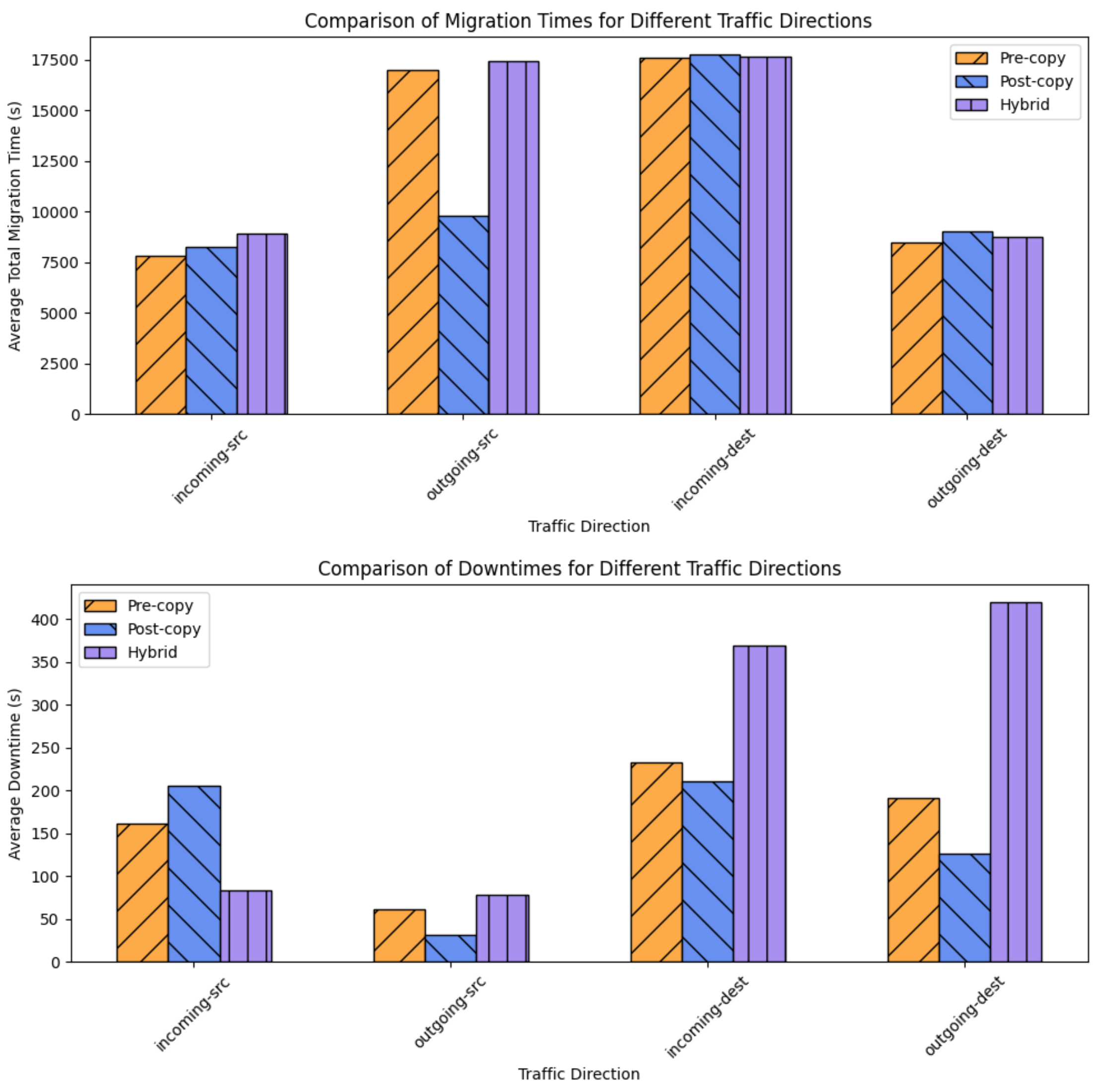


Figure 3: Total migration time and downtime with different traffic types

2. How can we efficiently migrate VMs in gang migration or serial migration by considering network contention and the migration deadlines to reduce migration time, downtime, and application performance degradation?

3. Can the integration of the hybrid copy technique enhance the efficiency of live gang migration when considering traffic contention?

2.4 Scope

2.4.1 In Scope

- Live Migration of VMs
  - The focus is on live migration within a KVM/QEMU virtualization environment.
  - Single NIC supported server with a single shared port for migration and application traffic. Some servers use separate NIC interfaces for the application and migration traffic (Fernando, Terner, et al. 2019). In these situations, the migration does not cause any contention with the business traffic. However, this condition is quite rare.
  - The host OS used for the servers is Linux.
  - Includes the live migration of a single VM from one host to another.
  - Covers scenarios involving the simultaneous live migration of multiple VMs between hosts.
  - Tests are carried out currently for LAN environments. These techniques can be adapted to WAN environments (disk migrations need to be considered).
- Analyzing the effect of traffic on live VM migration.
- Developing an optimized algorithm for traffic-sensitive live migration of multiple co-located VMs considering the priority of the migration.

- Evaluating migration performance using industry-expected benchmarks for migration time, downtime and application performance degradation.

2.5 Aims and Objectives

2.5.1 Aim

To develop an adaptive algorithm for traffic-sensitive live migration of multiple VMs that is migration priority aware, and optimizes overall network performance.

2.5.2 Objectives

- Identify key network metrics that affect VM migration performance.
- Create an adaptive decision-making algorithm that uses network metrics and migration deadlines to optimize live multiple VM migration.
- Investigate the impact of the hybrid-copy approach on the adaptability of live gang migration techniques.

2.6 Significance of the Research

This research is significant as it aims to develop an adaptive algorithm for traffic-sensitive live migration of multiple VMs, which is both migration priority aware and optimized for overall network performance. The focus is on making informed decisions about whether to use serial or parallel migration while also addressing the network contention problem.

Additionally, investigating the hybrid-copy approach will provide insights into enhancing the adaptability of live gang migration techniques. Ultimately, this research aims to reduce migration time, downtime, and application performance degradation, thereby improving service provisioning and communication resource efficiency.

3 Design and Methodology

3.1 Research Approach and Methodology

The proposed research approach follows a structured and systematic exploration to address traffic-sensitive VM migration optimization using QEMU with vanilla pre-copy, post-copy, and hybrid-copy techniques.

The initial phase involves conducting a detailed study with experiments that investigate the effects of varying network traffic levels, both incoming and outgoing, on migration performance during the movement of a single VM under different conditions.

Next phase focuses on analyzing network-intensive workloads within VMs during live migration. Experiments will assess the impact of different network conditions and traffic loads on host servers. The objective is to understand how different levels and directions of network traffic affect migration performance using pre-copy, post-copy, and hybrid methods. Results will guide the identification of key network parameters and metrics critical to optimizing migration strategies.

Building upon the previous phases, this stage extends the analysis to multiple VM migrations, both in serial and gang configurations. Different combinations of VM and host traffic patterns will be studied to understand their effect on migration performance. This phase seeks to identify optimal strategies for managing network contention, minimizing total migration time, and reducing downtime when multiple VMs are migrated concurrently or sequentially.

As the next stage, experimenting with bandwidth reservation tools and techniques to explore methods for dynamically adjusting bandwidth allocation during VM migration is done. The goal is to develop techniques that vary bandwidth according to user requirements and real-time network conditions to improve migration efficiency.

The following phase entails studying various types of workloads commonly seen in cloud environments, focusing on those most relevant to the study. Different types of workloads, their network usage characteristics, SLA considerations, and performance standards will be evaluated to ensure diverse testing scenarios.

In the implementation phase an algorithm will be developed to optimize the selection between serial and parallel migration strategies and to dynamically select the most suitable migration techniques based on real-time traffic patterns and migration urgency requirements. This priority-aware and traffic-sensitive algorithm will prioritize efficiency and minimal disruption.

The research follows the Design Science Research (DSR) methodology (Vom Brocke, Hevner, and Maedche 2020). The problem identification and motivation phases have been completed, defining objectives for optimizing VM migration using traffic-aware techniques. The next step is implementing the proposed optimization algorithm. This will be evaluated by comparing its performance against baseline models (vanilla pre-copy, post-copy, and hybrid migration) using metrics such as total migration time and downtime. Finally, the findings will be published to the research community.

3.2 Research Design

The proposed architecture for traffic-aware live VM migration, depicted in Figure 4, integrates several key components designed to optimize the migration process by monitoring and responding to network traffic conditions. Central to the architecture is the resource tracker, which monitors network traffic for both VMs and host machines before migration begins. This considers network conditions when making migration decisions. The network usage of individual VMs is tracked using *iptables*, where firewall rules are configured to monitor packet transfers between the host's NIC and relevant TAP devices associated with the VMs (Deshpande and Keahey 2017). For monitoring host network usage, the Linux utility *ifconfig* is employed.

The *migration controller* manages the selection and execution of the optimal migration technique between pre-copy, post-copy, hybrid-copy techniques and serial and parallel migrations, dynamically adapting based on network traffic contention and the VM migration's assigned priority.

To further enhance the efficiency of live VM migration, the *bandwidth reservation module* allocates network bandwidth for migrations on both the source and destination servers. This allocation is determined based on the priority of each migration, ensuring that higher-priority migrations receive the necessary network resources.

Both the source and destination hosts are connected through a TCP connection, and the entire system runs on top of the QEMU/KVM virtualization platform.

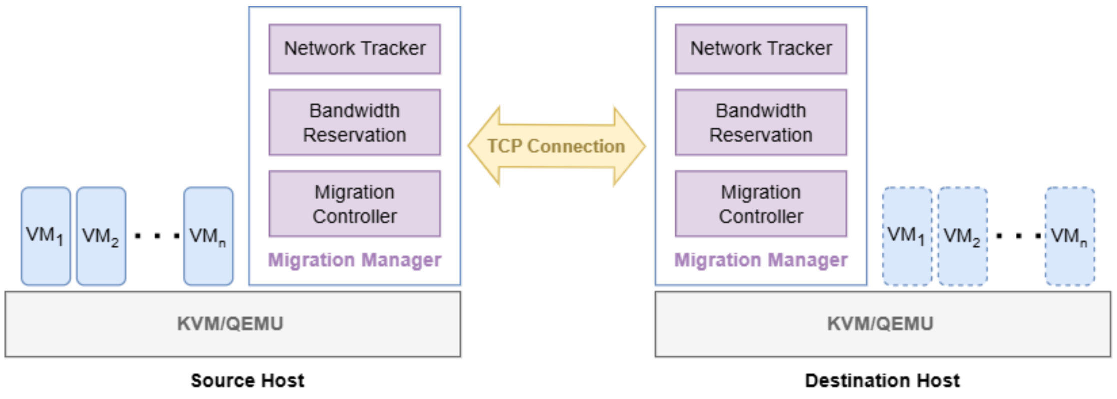


Figure 4: Proposed high-level architecture

3.3 Preliminary Results and Discussion

We conducted experiments using two (2) machines configured as source and destination hosts (As shown in Figure 5, with the following specifications:

- Product - HP Z620 Workstation
- CPU - Intel(R) Xeon(R) CPU E5-1650 v2 @ 3.50GHz (12 Cores)
- Memory - 16GB
- Network - Gigabit Ethernet Switch

An Network File System (NFS) server was employed, sharing a directory between the source and destination servers. The VM hard disks were stored on this NFS server, meaning they were not migrated during the LAN migrations. The baseline

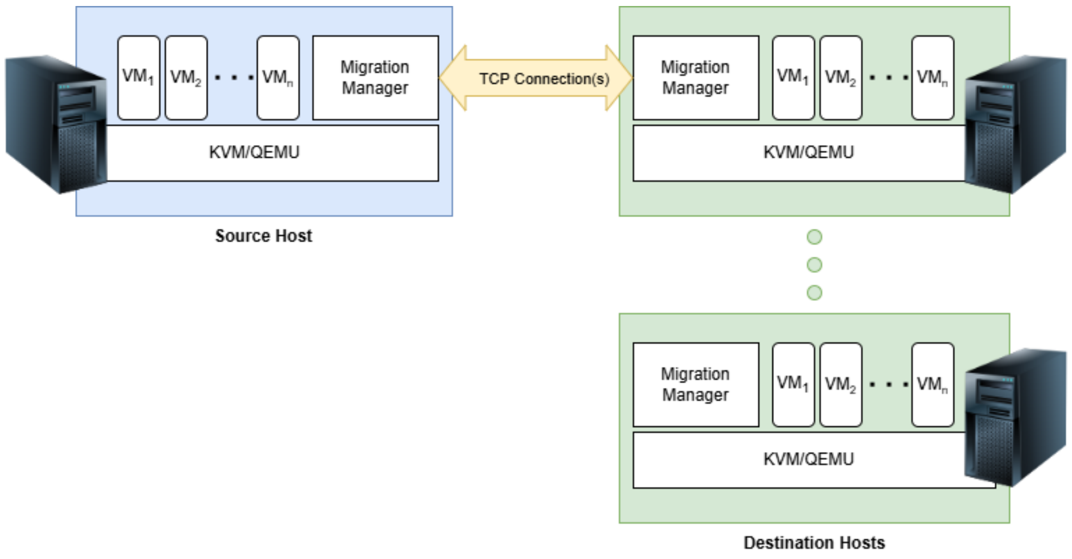


Figure 5: Testbed Architecture

models were implemented using QEMU Emulator version 2.5.0 installed on both the source and destination servers.

Single VM Migrations were performed using pre-copy, post-copy, and hybrid-copy methods under different traffic conditions to confirm the existence of the traffic contention problem and the results are shown in Figure 3. Higher migration times when there's outgoing traffic at the source and incoming traffic at the destination is due to contention of the migration traffic with network traffic.

- VMs - Idle VMs each with 8GB RAM.
- Traffic Combinations - Combinations of outgoing and incoming traffic at the source and destination.
- Measurement - Average values over ten (10) readings.

Multiple VM migration experiments were carried out to compare serial and parallel migrations under different traffic conditions. Two (2) VMs were initially started on the source host. A separate server acted as the iperf server and/or client, depending on the traffic combination being tested. Traffic was then generated between this third server and the source, destination, and VMs. Finally, the VMs were migrated to the destination server and the Total Migration Time and the Downtime were recorded. The results were averaged over five (5) readings, excluding any outliers. Figure 6 and figure 7 show the graphs plotted with the obtained results. The measurements taken will be used to identify the effect of traffic on total migration time and downtime and will be used when making migration decisions using a traffic-aware migration algorithm.

Total migration time in serial migrations is defined as the total time from the start of the first VM's migration until the completion of the second VM's migration. This represents the sum of both VMs' migration times, as the second migration begins as soon as the first completes. *Total Migration Time in parallel migrations* is the maximum migration time of the two VMs since both are migrated simultaneously. The total migration completes when the VM with the longest migration time finishes.

Downtime in serial migrations is the sum of the individual downtimes of each VM migration. *Downtime in parallel migrations* is the highest downtime among the individual migrations.

3.3.1 Monitoring VM Traffic

To monitor VM traffic, we measured the packets transferred by the VMs using *iptables.* This technique, used in studies by Cui, Zhu, et al. 2020 and Deshpande and Keahey 2017, involves creating firewall rules to capture traffic flow between the host's Ethernet interface (enp1s0) and the VM TAP interfaces (tap0 and tap1).

To validate the accuracy of these readings, we compared the packet counts obtained via *iptables* with those from the *ifconfig* command. The *ifconfig* readings showed an equal packet count to the sum from *iptables.*

The sum of the readings (R1 and R2) from *iptables* were equal to the total packets measured on the enp1s0 interface using *ifconfig.*

- *R1, R2* - Packet counts from *iptables* (traffic between tap0/tap1 and the host Ethernet interface)
- *enp1s0* - Packet count from *ifconfig* (total packets transferred and received on the Ethernet interface)

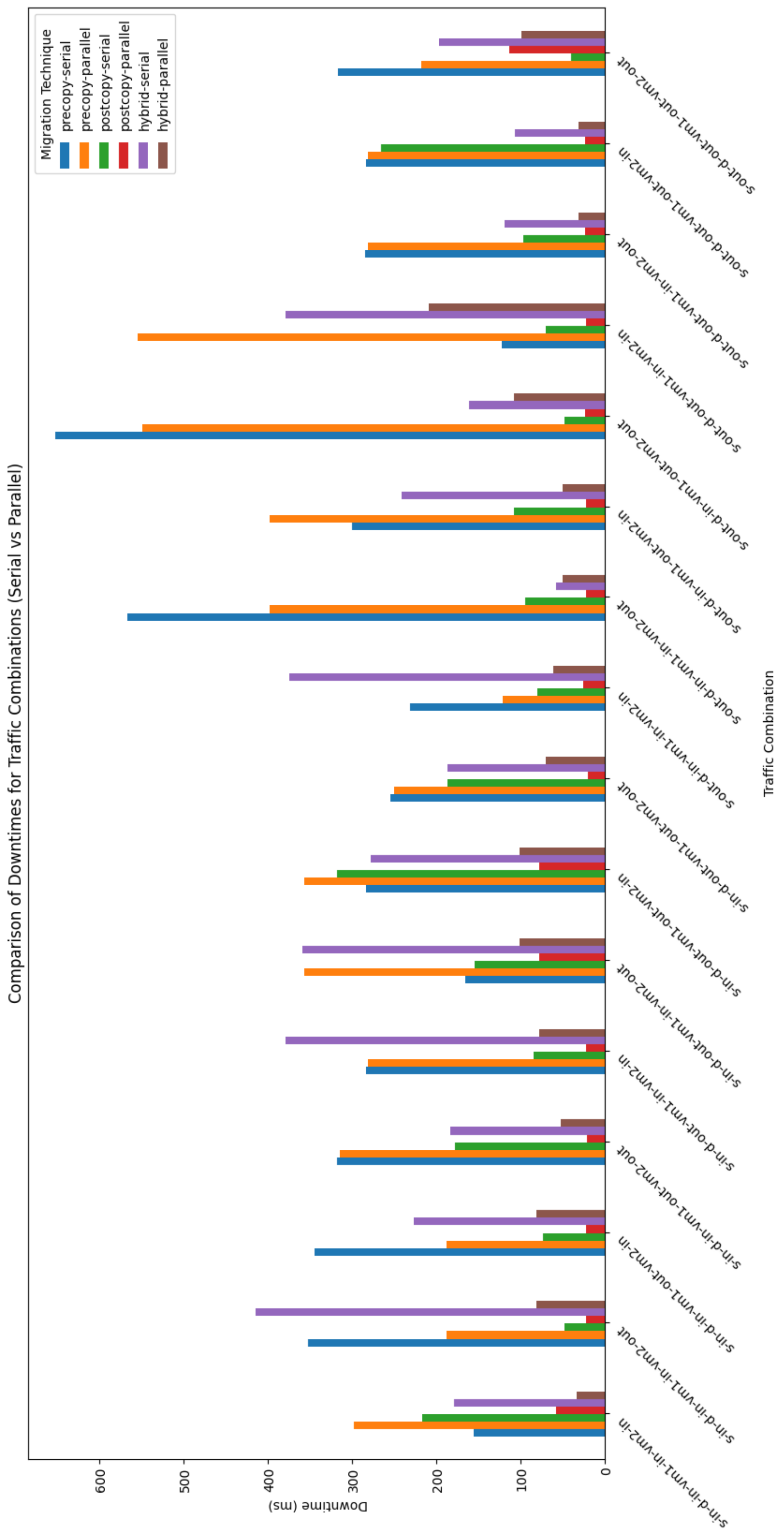


Figure 6: Total migration time of multiple VM migrations under different traffic types

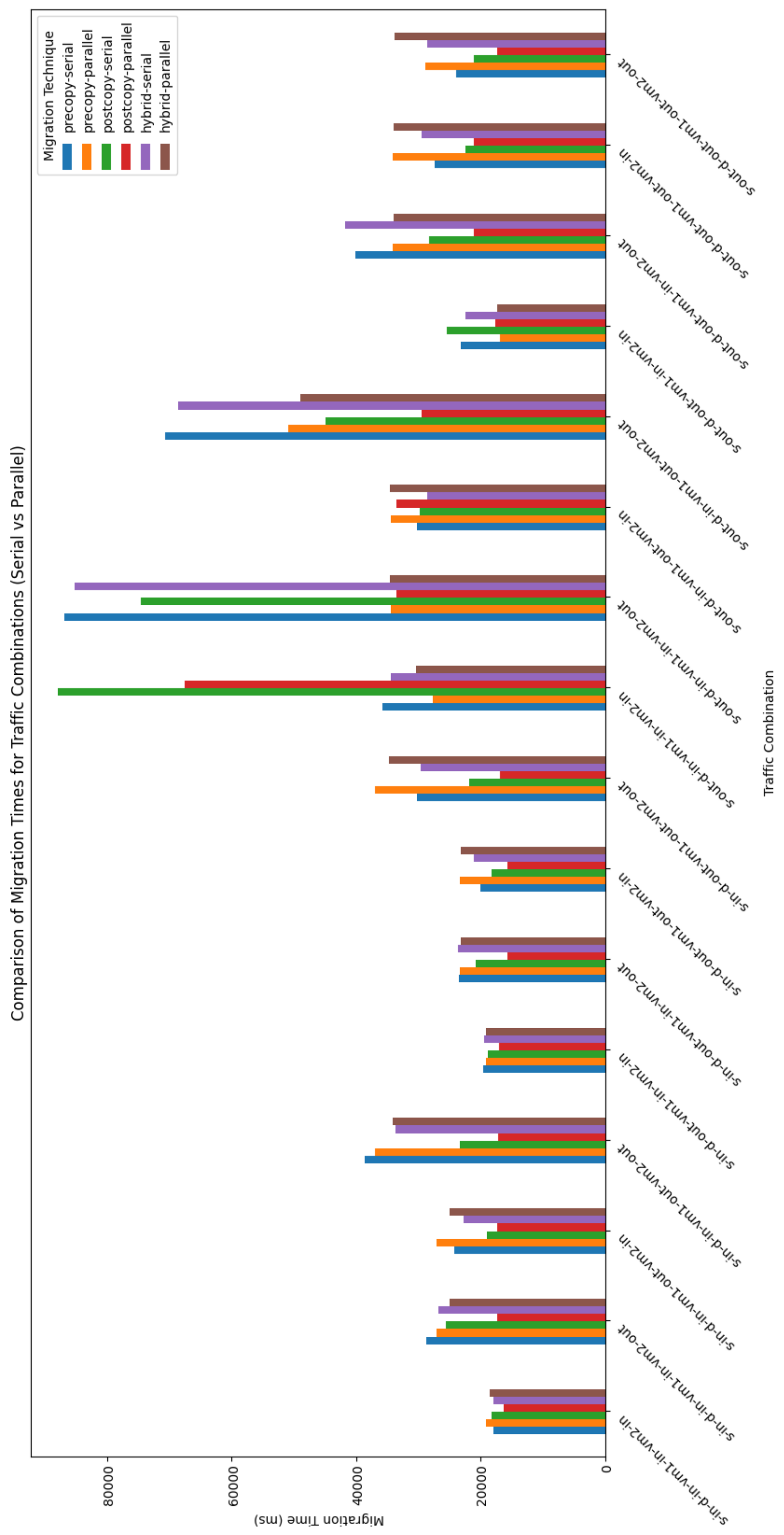


Figure 7: Downtime of multiple VM migrations under different traffic types

This confirms that both *iptables* and *ifconfig* are reliable tools for monitoring traffic for VMs, as well as source and destination servers.

3.4 Proposed Algorithm

Algorithm 1 Traffic-Aware VM Migration Algorithm

```
 1: procedure MigrationAlgorithm(VM_list, upper_limit, lower_limit)
 2:     Step 1: Cluster VMs
 3:         VM clusters ← Monitor and cluster VMs based on traffic
 4:     Step 2: Select migration technique
 5:         selected_techniques ← SelectMigrationTechniques(clusters)
 6:     Step 3: Determine and apply migration strategy
 7:     for each VM in VM list do
 8:         strategy ← SelectStrategy(contention, VM.priority)
 9:     end for
10:     Step 4: Reserve bandwidth and migrate VMs
11:         high_priority_vm_list, medium_priority_vm_list, low_priority_vm_list
        strategy ← Group VMs based on priority
12:         ReserveBandwidth(upper_limit, lower_limit, high_priority_vm_list,
        "high")
13:         ReserveBandwidth(upper_limit, lower_limit, medium_priority_vm_list,
        "medium")
14:         ReserveBandwidth(upper_limit, lower_limit, low_priority_vm_list, "low")
15:     Step 5: Finalize migration
16:         CompleteMigration
17: end procedure
```

3.4.1 Traffic-Aware VM Migration Algorithm

The proposed algorithm (Algorithm 1) optimizes live migration for multiple VMs based on real-time traffic conditions and priority levels while ensuring SLA compliance. The algorithm operates in five main stages.

Clustering of VMs Based on Traffic Patterns - The initial step clusters the VMs into groups according to their predominant traffic patterns, categorized as inbound or outbound.

Selection of Migration Technique - For each cluster, an optimal migration technique (pre-copy, post-copy, or hybrid-copy) is selected based on traffic conditions, including predominant traffic types at the source and destination, traffic contention levels, and the priority of the VMs. Traffic scenarios for the source and destination are categorized into *in-in, in-out, out-in,* and *out-out* whereas the traffic contention is divided into *high*, *moderate*, or *low* levels.

- High Contention and High Priority - If the traffic scenario is in-in or out-in for VMs in inbound clusters, post-copy migration is preferred to minimize downtime. For other scenarios, hybrid-copy is chosen to balance migration time and network load.
- High Contention and Low Priority - Hybrid-copy is selected as the default to reduce migration time without overloading the network.
- Moderate Contention - If the traffic scenario matches the cluster type (e.g., in-in for inbound clusters), pre-copy is prioritized for its lower network impact. Otherwise, hybrid-copy is used.
- Low Contention - Pre-copy migration is selected by default due to its efficiency under low-traffic conditions.

At this stage, each VM is updated in the *VM_list* with its assigned migration technique.

Assignment of Migration Strategy (Serial or Parallel) - Each VM is then assigned a migration strategy based on its priority and traffic contention level.

- High Priority VMs - Parallel migration
- Low Priority VMs with Low Contention - Parallel migration
- Low Priority VMs with High or Moderate Contention - Serial migration (to reduce network contention and potential performance degradation.)

This step ensures that each VM in the list has both a migration technique and a migration strategy assigned.

Bandwidth Reservation and Migration - The updated *VM_list* is split into three sublists: high-priority, medium-priority, and low-priority VMs. The *ReserveBandwidth* function then reserves the appropriate bandwidth for each group and carries out the migration in the following order: high-priority VMs first, followed by medium-priority, and finally low-priority VMs. The *ReserveBandwidth* function will be discussed later.

Once all VMs have been migrated, the migration process is finalized to ensure a smooth transition and minimal impact on overall system performance. This dynamic decision-making process ensures that the migration technique and strategy are optimized according to the current network conditions, migration priority and SLA requirements.

3.4.2 Bandwidth Reservation Algorithm

The *BandwidthReservation* algorithm (Algorithm 2) is designed to optimize bandwidth allocation during live VM migrations, with priority levels and SLA constraints taken into account. This algorithm dynamically adjusts bandwidth allocation for workload traffic and migration traffic based on migration priorities, ensuring that high-priority migrations receive sufficient bandwidth without causing SLA violations for ongoing workload traffic.

Input Parameters - The algorithm takes as input a list of VMs to be migrated, an upper limit and lower limit for workload traffic bandwidth (set by the system administrator to maintain SLA), and the priority level of the migration (high, medium, or low).

Capture Current Bandwidth Utilization - The current bandwidth utilization is measured using tools like iptables and ifconfig. The available bandwidth (unused bandwidth) is calculated, along with the workload bandwidth (the bandwidth currently used by workload traffic). The page transfer rate during migration, or transfer bandwidth, is estimated using QEMU.

Handling High-Priority Migrations - If the workload traffic consumes more bandwidth than the lower limit, the algorithm adjusts the workload traffic's

Algorithm 2 BandwidthReservation

```
1: procedure BandwidthReservation(upper_limit, lower_limit, VM_list, priority)
2:     Capture current_bw
3:     avail_bw ← $(1 - \text{current\_bw}/\text{tot\_bw}) \times 100\%$
4:     workload_bw ← $\text{current\_bw}/\text{tot\_bw} \times 100\%$
5:     transfer_bw ← page transfer rate during migration
6:     if priority = “high” then
7:         if workload_bw > lower_limit / tot_bw ×100% then
8:             reserve_workload_bw(lower_limit / tot_bw ×100%)
9:         end if
10:        Recapture current_bw; avail_bw ← $(1 - \text{current\_bw}/\text{tot\_bw}) \times 100\%$
11:        reserve_migration_bw(avail_bw)
12:        migrate_VMs(VM_list)
13:    else if priority = “medium” then
14:        if workload_bw > upper_limit / tot_bw ×100% then
15:            workload_bw ← upper_limit / tot_bw ×100%
16:        end if
17:        Recapture current_bw; avail_bw ← $(1 - \text{current\_bw}/\text{tot\_bw}) \times 100\%$
18:        reserve_migration_bw(avail_bw)
19:        migrate_VMs(VM_list)
20:        while !(migration_complete) do
21:            Recapture current_bw
22:            if current_bw < 100% and transfer_bw ≈ avail_bw then
23:                workload_bw ← current_bw - avail_bw
24:                if workload_bw < lower_limit then
25:                    reserve_migration_bw($(1 - \text{lower\_limit}/\text{tot\_bw}) \times 100\%$)
26:                else
27:                    reserve_migration_bw($(1 - \text{workload\_bw}/\text{tot\_bw}) \times 100\%$)
28:                end if
29:            end if
30:        end while
31:    else                                                ▷ priority = “low”
32:        reserve_migration_bw(avail_bw)
33:    end if
34:    migrate_VMs(VM_list)
35: end procedure
```

allocation to the minimum level necessary to satisfy SLA requirements (*lower_limit*). This step ensures maximum bandwidth is made available for high-priority migrations. After reallocating bandwidth for the workload traffic, the current bandwidth usage is recaptured, and the available bandwidth is recalculated. The entire unused bandwidth is then reserved for the migration of high-priority VMs, ensuring they complete quickly.

Handling Medium-Priority Migrations - If the workload traffic exceeds the upper limit, the algorithm reduces the workload bandwidth to the upper limit, freeing up additional bandwidth for migration. If the workload traffic is within the upper limit, no changes are made to its allocation. The current bandwidth usage is recaptured, and the available bandwidth is reserved for migration. While the migration is in progress, if there is a decrease in bandwidth usage (e.g., due to reduced workload traffic), the algorithm dynamically increases the bandwidth allocated for migration. However, the workload traffic's bandwidth allocation will not be reduced below the lower limit to ensure SLA compliance.

Handling Low-Priority Migrations - The algorithm reserves all available bandwidth for migration without making any changes to workload traffic allocation, as long as SLA compliance is maintained.

4 Evaluation Plan

4.1 Environment

All experiments will be conducted on servers with the following specifications: Product - HP Z620 Workstation; CPU - Intel(R) Xeon(R) CPU E5-1650 v2 @ 3.50GHz (12 Cores); Memory - 16GB RAM; Network - Gigabit Ethernet Switch; Operating System - Ubuntu 20.04 LTS; and Virtualization - QEMU Emulator version 2.5.0. The servers are interconnected via an HPE OfficeConnect 1920S Series Switch (JL385A). QEMU/KVM virtualization will be used to simulate live VM migrations.

4.2 Workload Generation

Workloads for the experiments will include both synthetic and real-world VM workloads to test migration performance under varied conditions.

1. **Transactional workloads -** Applications such as e-commerce and CRM require high availability, low latency, and strong data consistency (TPC-C, HammerDB).

2. **Analytical workloads -** Used for data lakes and business intelligence, these workloads need high throughput for large data sets (TPC-H, BigBench benchmarks will assess performance).

3. **Content Delivery Workloads -** Streaming media and CDNs demand high availability and low latency (Citron, wrk2)

4. **Development and Testing Workloads -** These have lower availability and moderate latency needs, with intermittent bursts (Phoronix Test Suite, Jenkins Plugins)

5. **Batch Processing Workloads -** Focused on tasks like ETL and report generation, these require cost efficiency and high throughput (HiBench, TPCx-BB).

6. **AI/ML Workloads -** Intensive tasks like model training need high scalability and throughput (MLPerf, DeepBench).

7. **Archival Workloads -** Long-term data storage prioritizes high durability with low bandwidth usage (SPEC SFS2014, fio).

This selection ensures diverse workload scenarios to test the effectiveness of the migration algorithm under real-world conditions.

4.3 Experimental Procedure

Initial experiments will be conducted using pre-copy, post-copy, and hybrid migration techniques, without any traffic-awareness mechanisms. This establishes a baseline against which traffic-sensitive optimizations can be measured. For each network-intensive workload type, randomized trials will be executed, with VMs migrated under varying network conditions. The Total Migration Time and Downtime will be measured and recorded for comparison. Repetition of experiments will ensure the reproducibility and robustness of results.

Next, the traffic-sensitive migration algorithm will be integrated into the QEMU virtualization environment. This component of the research will dynamically adapt migration techniques based on observed network traffic patterns and migration priorities.

Finally, the data analysis stage will focus on the following metrics Total Migration Time, representing the duration of the complete migration process for both traditional migration techniques and the traffic-sensitive migration algorithm, and Downtime, capturing the period during which a VM remains unavailable during migration. This metric will be compared across traditional techniques and the proposed algorithm. Statistical methods will be employed to analyze the Total Migration Time and Downtime across all experiments, quantifying performance improvements or degradations introduced by the traffic-sensitive migration strategy, with comparisons made to results from baseline models.

4.4 Project Timeline

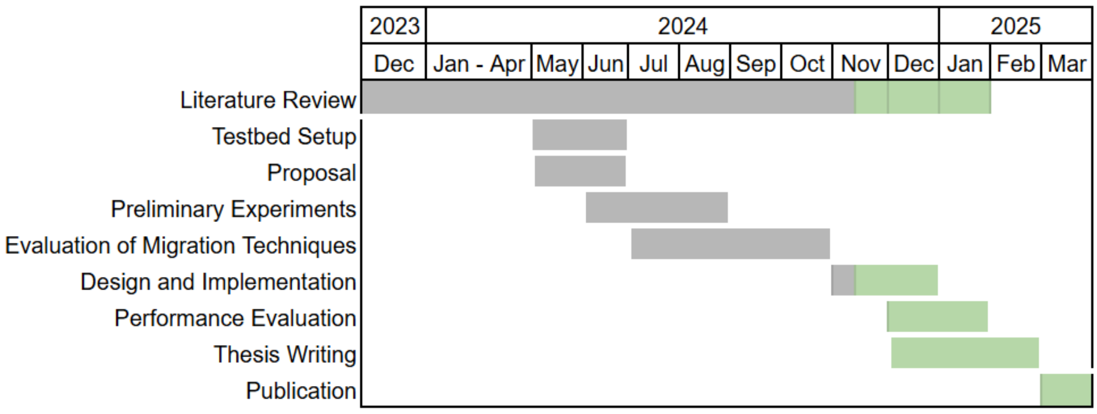


Figure 8: Project Timeline

References

Altahat, Mohammad A. et al. (2020). “Dynamic Hybrid-copy Live Virtual Machine Migration: Analysis and Comparison”. In: vol. 171. DOI: 10.1016/j.procs.2020.04.156.

Bahrami, Marziyeh, Abolfazl Toorooghi Haghighat, and Majid Gholipoor (n.d.). *A Review of Virtual Machine Migration Techniques in Data Center*. URL: https://www.researchgate.net/publication/372409749.

Callegati, Franco and Walter Cerroni (2013). “Live migration of virtualized edge networks: Analytical modeling and performance evaluation”. In: *2013 IEEE SDN for future networks and services (SDN4FNS)*. IEEE, pp. 1–6.

Cerroni, Walter and Flavio Esposito (2016). “Optimizing live migration of multiple virtual machines”. In: *IEEE Transactions on Cloud Computing* 6.4, pp. 1096–1109.

Chang, Victor, Robert John Walters, and Gary Wills (2013). “Cloud Storage and Bioinformatics in a private cloud deployment: Lessons for Data Intensive research”. In: *Cloud Computing and Services Science: Second International Conference, CLOSER 2012, Porto, Portugal, April 18-21, 2012. Revised Selected Papers 2*. Springer, pp. 245–264.

Choudhary, Anita et al. (2017). “A critical survey of live virtual machine migration techniques”. In: *Journal of Cloud Computing* 6.1, pp. 1–41.

Clark, Christopher et al. (2005). “Live migration of virtual machines”. In: *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation-Volume 2*, pp. 273–286.

Cui, Yong, Zhenjie Yang, et al. (2017). “Traffic-aware virtual machine migration in topology-adaptive dcn”. In: *IEEE/ACM Transactions on Networking* 25.6, pp. 3427–3440.

Cui, Yong, Liang Zhu, et al. (2020). “An adaptive traffic-aware migration algorithm selection framework in live migration of multiple virtual machines”. In:

International Journal of Performability Engineering 16 (2), pp. 314–324. ISSN: 09731318. DOI: 10.23940/ijpe.20.02.p14.314324.

Dalvandi, Aissan, Mohan Gurusamy, and Kee Chaing Chua (2015). "Time-aware vmflow placement, routing, and migration for power efficiency in data centers". In: *IEEE Transactions on Network and Service Management* 12.3, pp. 349–362.

Deshpande, Umesh and Kate Keahey (July 2017). "Traffic-sensitive Live Migration of Virtual Machines". In: *Future Generation Computer Systems* 72, pp. 118–128. ISSN: 0167739X. DOI: 10.1016/j.future.2016.05.003.

Deshpande, Umesh, Xiaoshuang Wang, and Kartik Gopalan (2011). "Live gang migration of virtual machines". In: *Proceedings of the 20th international symposium on High performance distributed computing*, pp. 135–146.

Devi, L Yamuna, P Aruna, N Priya, et al. (2011). "Security in virtual machine live migration for KVM". In: *2011 International Conference on Process Automation, Control and Computing.* IEEE, pp. 1–6.

Fernando, Dinuni, Jonathan Terner, et al. (2019). "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.* IEEE, pp. 343–351.

Fernando, Dinuni, Ping Yang, and Hui Lu (2020). "SDN-based Order-aware Live Migration of Virtual Machines". In: *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, pp. 1818–1827. DOI: 10.1109/INFOCOM41043.2020.9155415.

Fu, Xiong et al. (2019). "Network traffic based virtual machine migration in cloud computing environment". In: *2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC).* IEEE, pp. 818–821.

Gao, Yongqiang et al. (2014). "Service level agreement based energy-efficient resource management in cloud data centers". In: *Computers & Electrical Engineering* 40.5, pp. 1621–1633.

Haidri, Raza A et al. (2022). “A deadline aware load balancing strategy for cloud computing”. In: *Concurrency and Computation: Practice and Experience* 34.1, e6496.

Hines, Michael R, Umesh Deshpande, and Kartik Gopalan (n.d.). *Post-Copy Live Migration of Virtual Machines.*

Hines, Michael R and Kartik Gopalan (2009). “Post-copy based live virtual machine migration using adaptive pre-paging and dynamic self-ballooning”. In: *Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*, pp. 51–60.

Hu, Liting et al. (2008). “Magnet: A novel scheduling policy for power reduction in cluster with virtual machines”. In: *2008 IEEE International Conference on Cluster Computing.* IEEE, pp. 13–22.

Huang, Wei et al. (2007). “Virtual machine aware communication libraries for high performance computing”. In: *Proceedings of the 2007 ACM/IEEE conference on Supercomputing*, pp. 1–12.

Jo, Changyeon et al. (2013). “Efficient live migration of virtual machines using shared storage”. In: *ACM Sigplan Notices* 48.7, pp. 41–50.

Kanniga Devi, R, G Murugaboopathi, and M Muthukannan (2018). “Load monitoring and system-traffic-aware live VM migration-based load balancing in cloud data center using graph theoretic solutions”. In: *Cluster Computing* 21.3, pp. 1623–1638.

Kherbache, Vincent, Eric Madelaine, and Fabien Hermenier (2017). “Scheduling live migration of virtual machines”. In: *IEEE transactions on cloud computing* 8.1, pp. 282–296.

Kuno, Yosuke, Kenichi Nii, and Saneyasu Yamaguchi (2011). “A study on performance of processes in migrating virtual machines”. In: *2011 Tenth International Symposium on Autonomous Decentralized Systems.* IEEE, pp. 567–572.

Le, Tuan (Nov. 2020). *A survey of live Virtual Machine migration techniques.* DOI: 10.1016/j.cosrev.2020.100304.

Liu, Jiaqiang et al. (2015). "Traffic aware cross-site virtual machine migration in future mobile cloud computing". In: *Mobile Networks and Applications* 20, pp. 62–71.

Lu, Kuan et al. (2013). "QoS-aware VM placement in multi-domain service level agreements scenarios". In: *2013 IEEE Sixth International Conference on Cloud Computing*. IEEE, pp. 661–668.

Maheshwari, Sumit et al. (2018). "Traffic-aware dynamic container migration for real-time support in mobile edge clouds". In: *2018 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*. IEEE, pp. 1–6.

Manzalini, Antonio et al. (2013). "Clouds of virtual machines in edge networks". In: *IEEE Communications Magazine* 51.7, pp. 63–70.

Nadeem, Hanan A, Hanan Elazhary, and Mai A Fadel (2018). "Priority-aware virtual machine selection algorithm in dynamic consolidation". In: *International Journal of Advanced Computer Science and Applications* 9.11.

Nagarajan, Arun Babu et al. (2007). "Proactive fault tolerance for HPC with Xen virtualization". In: *Proceedings of the 21st annual international conference on Supercomputing*, pp. 23–32.

Nasim, Robayet and Andreas J Kassler (2015). "Network-centric performance improvement for live VM migration". In: *2015 IEEE 8th International Conference on Cloud Computing*. IEEE, pp. 106–113.

Nathuji, Ripal and Karsten Schwan (2007). "Virtualpower: coordinated power management in virtualized enterprise systems". In: *ACM SIGOPS operating systems review* 41.6, pp. 265–278.

Nelson, Michael, Beng-Hong Lim, Greg Hutchins, et al. (2005). "Fast Transparent Migration for Virtual Machines." In: *USENIX Annual technical conference, general track*, pp. 391–394.

Odun-Ayo, Isaac, Blessing Udemezue, and Abiodun Kilanko (2019). "Cloud service level agreements and resource management". In: *Adv. Sci. Technol. Eng. Syst.* 4.2, pp. 228–236.

Padala, Pradeep et al. (2007). “Adaptive control of virtualized resources in utility computing environments”. In: *Proceedings of the 2nd ACM SIGOPS/EuroSys European Conference on Computer Systems 2007*, pp. 289–302.

Sahni, Shashank and Vasudeva Varma (2012). “A hybrid approach to live migration of virtual machines”. In: *2012 IEEE international conference on cloud computing in emerging markets (CCEM)*. IEEE, pp. 1–5.

Shrivastava, Vivek et al. (2011). “Application-aware virtual machine migration in data centers”. In: *2011 Proceedings IEEE INFOCOM*. IEEE, pp. 66–70.

Son, Jungmin and Rajkumar Buyya (2018). “Priority-aware VM allocation and network bandwidth provisioning in software-defined networking (SDN)-enabled clouds”. In: *IEEE Transactions on Sustainable Computing* 4.1, pp. 17–28.

Soni, Gulshan and Mala Kalra (Dec. 2013). “Comparative Study of Live Virtual Machine Migration Techniques in Cloud”. In: *International Journal of Computer Applications* 84 (14), pp. 19–25. DOI: 10.5120/14643-2919.

Sun, Gang, Dan Liao, Vishal Anand, et al. (2016). “A new technique for efficient live migration of multiple virtual machines”. In: *Future Generation Computer Systems* 55, pp. 74–86.

Sun, Gang, Dan Liao, Dongcheng Zhao, et al. (2015). “Live migration for multiple correlated virtual machines in cloud-based data centers”. In: *IEEE Transactions on Services Computing* 11.2, pp. 279–291.

Tsakalozos, Konstantinos et al. (2017). “Live VM migration under time-constraints in share-nothing IaaS-clouds”. In: *IEEE Transactions on Parallel and Distributed Systems* 28.8, pp. 2285–2298.

Tso, Fung Po et al. (2014). “Scalable traffic-aware virtual machine management for cloud data centers”. In: *2014 IEEE 34th International Conference on Distributed Computing Systems*. IEEE, pp. 238–247.

Vom Brocke, Jan, Alan Hevner, and Alexander Maedche (2020). “Introduction to design science research”. In: *Design science research. Cases*, pp. 1–13.

Voorsluys, William et al. (2009). *Cost of Virtual Machine Live Migration in Clouds: A Performance Evaluation*. URL: http://www.cloudbus.org.

Wood, Timothy, KK Ramakrishnan, et al. (2014). "CloudNet: Dynamic pooling of cloud resources by live WAN migration of virtual machines". In: *IEEE/ACM Transactions On Networking* 23.5, pp. 1568–1583.

Wood, Timothy, Prashant J Shenoy, et al. (2007). "Black-box and Gray-box Strategies for Virtual Machine Migration." In: *NSDI*. Vol. 7, pp. 17–17.