A Review: Live VM Migration Convergences in a Dynamic Workload

Literature Review
SCS 3216 : Research Methods

Nadeesha Epa
Index No:20000499

May 3, 2025

Declaration

The literature review is my original work and has not been submitted previously for any examination/evaluation at this or any other university/institute. To the best of my knowledge, it does not contain any material published or written by another person, except as acknowledged in the text.

Student Name : D.N.N.Epa
Registration Number : 2020/cs/049
Index Number :20000499

_____

Signature & Date

This is to certify that this literature review is based on the work of Ms. D.N.N.Epa under my supervision. The literature review has been prepared according to the format stipulated and is of an acceptable standard.

Supervisor Name : Dr.Dinuni Fernando

_____

Signature & Date

Acknowldegment

I must thank my supervisor Dr. Dinuni K. Fernando for guiding and providing valuable advice and guidance in completing this review. I also thank Dr. Manjusri Wickramasinghe for giving me the domain knowledge for better research during the Research Methods Course module. Finally, I would like to thank my parents family, and friends for helping and motivating me to complete this review.

Abstract

As technology grows the requirement for more efficient and reliable applications has become a must. A key problem that had previously been there was not enough storage for several applications. The Solutions that fulfill these requirements are called Cloud Computing and Virtualization. With the use of Cloud Computing users were able to create and customize their application needs using the internet and Virtualization allows users to generate a virtual representation of actual hardware.

With the use of these technologies the application(VM) performance has been optimized while the problems regarding storage and maintenance have been solved. However, the failures can happen at any time. In order to avoid the interruptions happening because of the failures the VMs will be migrated to another server. This process is introduced as the VM migration. One of the key terms that need to be considered before the VM migration is identifying the convergence point to start the migration process. According to the nature of the workload, the convergence point will be changed. For a static workload, it is very easy to identify the convergence point. However, if the workload is dynamic then additional effort has to be put to handle the dynamic nature. Machine learning-based methods and Profiling can be defined as two(2) main types of techniques used to handle a dynamic workload.

The report starts the discussion by explaining the terms Cloud Computing, Virtualization, VM migration, and Dynamic workloads. Then a detailed description of the different types of dynamic workload handling methods with their advantages and limitations are discussed. Finally, the report explains the existing dynamic workload migration techniques and improvements that can be made to improve the quality of the migration process. The report concludes by proposing a new method that combines both machine learning techniques and profiling.

Keywords: Virtualization, Cloud Computing, Virtual Machines, Dynamic workload, Migration, Convergence

Contents

List of Figures

List of Tables

Acronyms

CC Cloud Computing. 1, 16, 21

CDC Cloud Data Center. 1

CPU Centrel Processing Uniy. 3, 6

CRM Cloud Resource Manager. 16, 18, 20

CUA Capacity and Utility Agent. 11

DC Data Centers. 10

DQ Dynamic Quantum. 12

IaaS Infrastructure as a service. 12

IO Input Output. 3, 6

ML Machine Learning. 10–12, 22

RTM Real Time Monitoring. 11

SaaS Software as a service. 12

TMT Total Migration Time. 6, 8, 9, 15

VM Virtual Machine. iv, 2–8, 14–18, 20

VMs Virtual Machines. 4, 9, 17

WWS Writable working set. 5, 14

1 Introduction

1.1 Cloud Computing and Virtualization

As technology grows the requirement for more efficient and reliable applications has become a must. A key problem that had previously been there was not enough storage for several applications. Over the previous years, several researches have been conducted to find different technologies to increase the efficiency of applications and find more storage for applications. As a result, researchers were able to find two(2) technologies called Cloud Computing(CC)(Malhotra et al. 2014)(Gong et al. 2010) and Virtualization(Xing & Zhan 2012). Virtualization can also be known as one of the key concepts used in CC. This combination brings more efficiency and scalability to cloud users. CC not only provides storage facilities to users but also consists of different types of services such as hardware, software, infrastructure, and security.

1.1.1 Cloud Computing

With the use of CC users were able to create and customize their application needs using the internet. So it provides the ability for users to access data stored on servers from anywhere in the world. Because of its flexibility and cost-effectiveness, it has been popular all around the world. Google Cloud(GCP 2024), Amazon Web Services (AWS)(AWS 2024), Microsoft Azure(Microsoft 2024), and Oracle(Oracle 2024) are among the top ten(10) cloud service providers. Cloud Data Centers(CDC) are responsible for providing resources that are needed for applications to execute. The collection of these application resources that are available in the cloud is simply known as the cloud workload(CW).(Kashyap & Singh 2023) provides four(4) main types of cloud workload applications as **Business-critical workload applications, Mission-critical workload applications, Low-priority workload applications, and Scientific computing workload applications.** Users can select the most suitable application type depending on their workload requirements. Even though the CC consists of several advantages, it has a few challenges as well. Handling the Dynamic workload behavior, understanding the resource demands before distributing them and the need for efficient scheduling are some of them. To handle these situations different technologies have been proposed by several researchers(Lin et al. 2011),(Sah & Joshi 2014),(Banerjee et al. 2023) which will be discussed more in section 3.3.

1.1.2 Virtualization

As the name suggests Virtualization (Xing & Zhan 2012)(Malhotra et al. 2014) defines the process of generating a virtual representation of actual hardware. This allows users to use the functionality of multiple hardware resources by placing multiple virtual machines inside one physical machine. With the use of Virtualization, it removes limitations such as the need for electricity for physical servers, storage problems, and maintenance problems. Virtual machine(VM) and Hypervi-

sor sometimes known as the Virtual Machine Manager(VMM) can be introduced as the two important concepts used in Virtualization. VM acts as a software-defined computer that runs inside a physical computer. But the advantage here is this VM consists of its own operating system and computing resources. The Hypervisor is responsible for managing multiple virtual machines in a computer. According to the requirements of the users they can use different types of virtualization to accomplish their needs. **Server Virtualization, Network Virtualization, Storage Virtualization, and Data Virtualization**(Mohammed & Kiran 2014) are some of them. After deciding the actual type of virtualization user can use virtualization software to implement the specific type of virtualization. SolarWinds Virtualization Manager (SolarWinds Worldwide, LLC 2024), V2 Cloud (V2 Cloud Solutions, Inc 2024), Oracle VM Virtualbox (Oracle 2024), and Red Hat Virtualization (Red Hat, Inc 2024) are within the top ten(10) Virtualization Software that is available in the market.

1.2 VM Migration

VM Migration define the idea of transferring a VM running on one resource(Source) to another resource(Destination). This resource can be another physical machine or a data store. Since the failures of a VM can happen at any time migration is essential for low-level system maintenance, load balancing, and fault management Choudhary et al. (2017). According to the papers (Fernando et al. 2020) and (Miller 2008), it has shown that the normal failure rate of Google data centers is 1000 machine failures for each cluster's first year. Hence in the instance of a failure, the VM will be quickly migrated to avoid interruptions for users who use the services provided by the failed VM. There are primarily two(2) kinds of VM migration techniques as **Live migration(Hot migration)**Nayyer et al. (2019) and **Non-live migration(Cold migration)**.

1.2.1 Live Migration

During the middle of execution, there can be failures. This is the instance where the need for live migration comes into play. For a better user experience, the need for live migration is essential. (Christopher 2005) has shown several advantages of live migration as

- In live migration the entire Operation System and all of its apps are moved at once. This will avoid many challenges encountered by process-level migration methodologies. (Christopher 2005)
- After the migration the host machine does not depend on the source and this solves the problem of residual dependencies.
- Live VM migration has the ability to complete the migration process without providing an interruption to the users.

Live VM migration can be categorized broadly into three(3) main techniques as **Pre-Copy migration, Post-Copy migration, and Hybrid migration**(He

2021). These techniques will be further explained in section 2.

1.2.2 Non-live Migration

Non-live migration(Hines et al. 2009) happens when the VM is powered off. First, the source VM paused its execution and quickly transferred the current memory and all the configurations to the destination VM . Then VM continues its execution at the destination. This technique has resulted in a downtime which affects the services executed during the migration. However, non-live migration has several advantages. During the non-live migration, there is no need to handle the execution of a running machine while sending its state to the destination. Hence non-live migration becomes more reliable compared to live migration. Another advantage is that if the VM that needs to be migrated has a heavy memory load, then the non-live migration will be more suitable than live migration because migrating such VM using live migration can be challenging due to the bandwidth and resource demands.

1.3 Performance Metrics

The use of a performance matrix is to measure how efficiently the migration process is completed. (Galloway et al. 2015) and Strunk & Dargie (2013) provide a study about the metrics that are used to measure performance such as **Migration time, CPU utilization, RAM utilization, Network traffic, and Downtime**. The definitions for those metrics are as below.

- Migration time - Duration between transferring the VM from source to destination(Alamdari & Zamanifar 2012)
- CPU utilization - Measure the CPU resources consumed by both source and destination hosts(Strunk & Dargie 2013)
- RAM utilization - Measure the RAM consumed by both source and destination hosts(Strunk & Dargie 2013)
- Network traffic - Measure the total volume of data that is moved during the migration.(Strunk & Dargie 2013)
- Downtime - Measures the amount of time VM paused its execution in the middle of migration due to the transferring of CPU and IO states.(Strunk & Dargie 2013)

For better performance, it is crucial to minimize these factors as much as possible. Several Optimizations have been applied for migration techniques to reduce the factors discussed above and those optimizations will be discussed in detail in section 2.3.

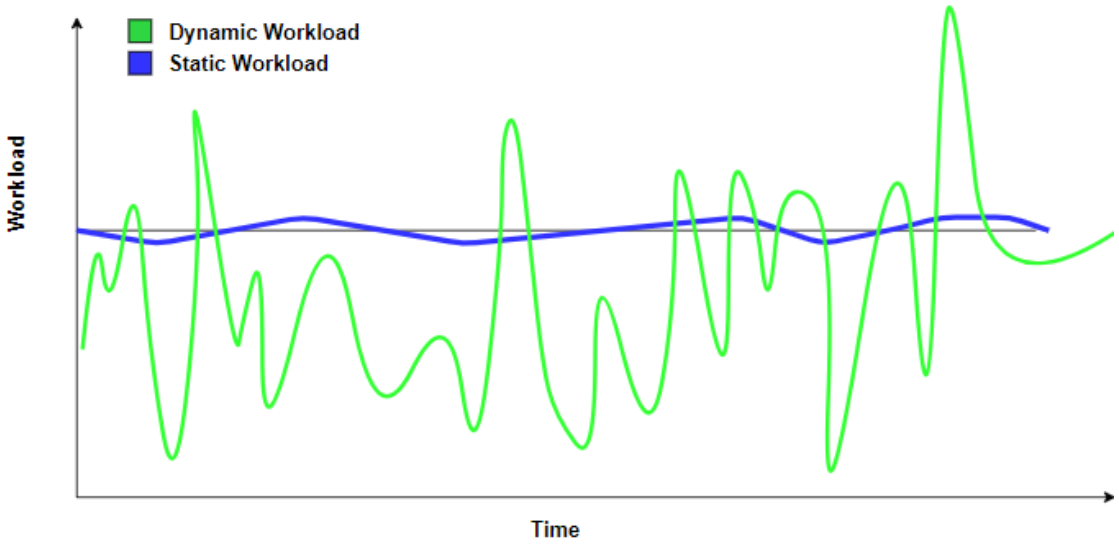

Figure 1: Dynamic workload vs Static workload

1.4 Workload Migration

Workload migration refers to the term migrating a workload from one environment to another environment. As discussed in the section 1.1.2 VMs needs to be migrated to avoid the interruptions that occur due to the failures of VMs. A successful migration requires the transferring of source VM to the destination VM. However, the selection of migration technique depends on the nature of the workload. Workloads can be mainly categorized into two sections dynamic workloads and static workloads . Static workload consists of a constant amount of computing resources over a long period and because of this nature, it is easy to predict the behavior of the workload and start the migration process. While the dynamic workload changes over time it is hard to predict its behavior by considering the current workload patterns. So different techniques need to handle a dynamic workload. These techniques will be discussed in detail in section 3.3. Figure 1 shows the comparison between the dynamic workload and the static workload.

2 Workload Migration Techniques

In this section, different types of live migration techniques, implementation details for those techniques, and their convergence in terms of the static and dynamic nature of the residing workload will be discussed in detail.

2.1 Live Migration Techniques

As previously mentioned the three(3) main sections of live migration techniques are Pre-Copy , Post-Copy, and Hybrid migration . They differ from each other in the way that they are used to **handle the memory** and **transfer the CPU state.**

2.1.1 Pre-Copy Migration

In Pre-Copy(Christopher 2005),(Fernando et al. 2020) migration the memory transferring will happen via 3 main steps.

1. Push phase
   During the push phase memory pages will be sent to the destination while the source VM continues its execution. This process executes continuously for several iterations. On the first iteration, each stored memory page will move from the source to the destination. In the second iteration, only the modified memory pages compared to the first iteration will be transferred. Likewise, for the next iteration, only the modified pages compared to the previous iteration will be transferred.
   If the workload is **static** then the required transferring page count for the coming iterations will become low compared to the first iterations. Then at some point, there is an instance that the same set of pages will become dirty for several iterations. This set of dirty pages will be identified as the **writable working setWWS** . For the static workload migration, this point will be considered as the convergence point.
   If the workload is **dynamic** then there is no guarantee that there is a point where after that point the dirty page count will be reduced. In there the working set will change repeatedly during the migration. Hence it is really hard to identify a convergence point. So, migrating a dynamic workload needs special types of mechanisms such as profiling (Ye et al. 2014) and machine learning-based methods(Wei et al. 2018). More details about these techniques will be discussed in section 3.3.

2. Stop and copy phase
   After identifying the convergence point the source VM holds its execution and quickly sends the remaining pages to the destination VM. From this point onwards the destination VM starts to be executed. The duration that source VM paused its execution and started resuming in the destination is known as the **downtime** of the VM migration.

3. Pull phase
   While the VM executes in the destination it may notice that some missing pages have not been successfully transferred to the destination. This instance is known as a **page fault**. In this case, the source will re-send the missing pages to the destination.

2.1.2 Post-Copy Migration

The execution nature of the Pre-Copy technique is used to send the dirty pages to the destination through multiple iterations. But in Post-Copy technique(Hines et al. 2009),(Fernando et al. 2020) migration it uses a mechanism that ensures no duplicates of the same page will be sent to the destination. The steps of this technique are as follows.

1. First, the VM at the source node paused its execution.Then minimal set of memory pages required for the execution of the destination VM will be moved to the destination.
2. Then the destination VM starts its execution and the source quickly starts sending the remaining memory pages to the destination. This process is known as **pre-paging**.
3. Same as the pull-phase in the Pre-Copy if the destination VM may notice that some pages have not been successfully sent to the destination a page fault will occur and the source quickly starts sending the missing pages to the destination. This is known as the **Demand paging**

2.2 Pre-Copy Vs Post-Copy

This section compares the performance of Pre-Copy and Post-Copy algorithms according to their preparation time, resume time, downtime, page transferred, and total migration time(Hines et al. 2009),(Galloway et al. 2015). Here are the definitions for the matrices that were used for comparison.

- **Preparation Time** - Duration between the point where the migration has been initialized and successfully moving the source VM's CPU state to the destination host.
- **Downtime** -Amount of time VM paused its execution in the middle of migration due to the transferring of CPU and IO states.
- **Resume time** - Amount of time takes to restart the VM in the destination. This is the point that is considered as the migration is over.
- **Page Transferred** - Total amount of memory pages transferred throughout the migration process.

Table 1 depicts the contrast between Pre-Copy and Post-Copy techniques. Both techniques have their own advantages as well as disadvantages. Pre-Copy performs well for a read-intensive workload(applications that involve a lot of reading data compared to writing or modifying it), but it doesn't perform well for write-intensive workloads. Write-intensive workloads result in a lot of dirty pages that proportionally increase the TMT . Post-copy migration performs well with the write-intensive workload and doesn't perform well for the read-intensive workloads. The reason behind this is that read-intensive workloads increase the number of page faults which results in a high response time(Sahni & Varma 2012) and the impact for the Post-Copy from the write-intensive workloads becomes low as the Post-Copy migration starts by sending the CPU and IO states to the destination. The comparison shows the need for another migration technique that works well for both read-intensive and write-intensive workloads.

Matrix	Pre-Copy Migration	Post-Copy Migration
Preparation time	Entire iterative memory copy phase	The preparation time is minimal because the CPU state has already been transferred when starting the migration process.
Downtime	Duration between transferring the CPU state and remaining dirty pages.	Time takes to move the memory pages required to up the destination VM
Resume Time	Duration taken to create the destination VM that can fully execute without the source VM.	Time takes for the pre-paging stage.
Page Transferred	The majority of the memory pages will sent at the preparation time.	The majority of the memory pages will sent at the response time.

Table 1: Pre-Copy vs Post-Copy

2.2.1 Hybrid Migration

Hybrid migration(Sahni & Varma 2012),(Altahat et al. 2020) can be known as a combination of both Pre-Copy and Post-Copy techniques. Hybrid migration aims to increase the performance for both read-intensive and write-intensive workloads. The algorithm consists of one(1) Pre-Copy round and the rest of the algorithm consists of the Post-Copy technique. Hybrid algorithm stages can be described as given below. Figure 2 explains more about the stages of hybrid migration.

1. As the first step the Pre-Copy round executes. This step is used to make a copy of the memory of the VM. During the execution of this first round, the source VM keeps running.
2. Then the source VM stopped and the CPU state of the source VM will transfer to the destination host.
3. As the final step the Post-Copy algorithm starts the execution and the remaining dirty pages will copy from the source to the destination. If any page fault occurs the page will be copied to the destination from the source(Demand Paging).

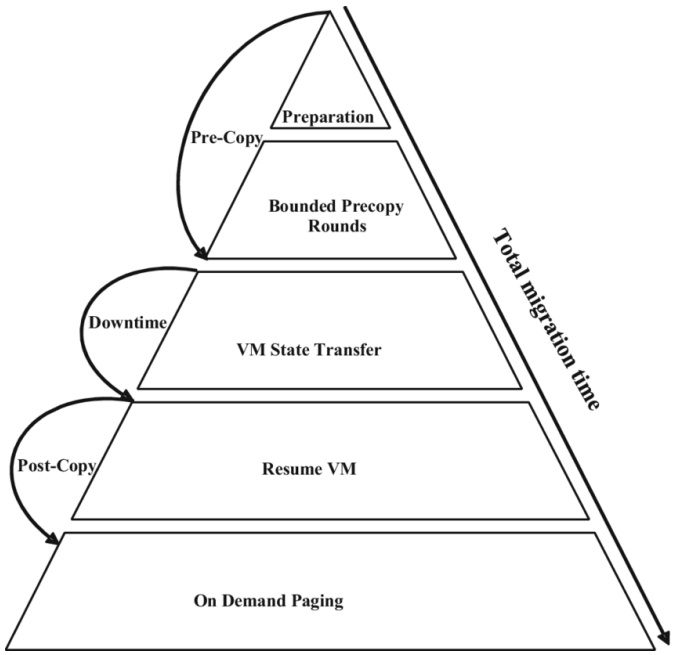

Figure 2: Stages of Hybrid migration
(Ahmad et al. 2015)

2.3 Migration Optimizations

As discussed earlier to increase the performance of the migration techniques several optimizations have been implemented. This section provides a brief description of some selected techniques such as implementation details of those techniques, and how those techniques can be used to increase performance.

- Quick Eviction Method (Fernando et al. 2016)
  In the Quick Eviction method, by using Live snapshots it reduces the TMT. The algorithm for this method was implemented by modifying the Pre-Copy algorithm. Quick Eviction uses a new dirty page tracking algorithm which works more efficiently than the normal Pre-Copy algorithm. The steps of the algorithm are briefly as below.
  - In the Quick Eviction method snapshots of the VM's memory will be transferred to a remote location before the migration starts.
  - When sending the snapshots the first iteration takes a snapshot of all the memory pages and for the next iterations only the snapshot of the modified pages compared to the previous iteration will be sent to the destination.
  - Once the migration starts the process has to only transfer the modified memory pages compared to the last snapshot.

  Since most of the pages are already in the destination host it helps to make the destination execute without depending on the source VM very quickly. This will lead to a shorter TMT.

- Software Defined Networking for Live Migration (Fernando et al. 2020)

In this method Software Defined Networking technique is used to provide a novel approach. The provided method has shown a significant reduction in the TMT when there are multiple VMs to be migrated. The idea behind this approach is to identify an order for VMs to migrate based on their resource contention. This method is known as **SOLive**. The steps of the algorithm are briefly as below.

- As the first step SoLive uses a resource tracking module to calculate the resource usage for each VM.
- Then based on the resource usage, the migration scheduler will create an order for VMs to migrate.
- SoLive also uses a bandwidth reservation module that can monitor the changes in the bandwidth and dynamically reserve the bandwidth suitable for the migration.

The paper(Fernando et al. 2020) has proven that this method has the ability to work well for both Pre-Copy and Post-Copy migration techniques to reduce the TMT.

- Statistical Prediction and Compression Model (Patel et al. 2018)
  This paper provides an improvement for the Pre-Copy algorithm using the prediction and compression methods. The method uses a prediction model called **ARIMA** to predict dirty pages and a delta compression method to handle the frequently updated pages. The combination of these two techniques was able to reduce the Downtime and the TMT of the migration process.

The above techniques can be introduced as a few techniques that use optimizations for migration techniques.

3 Dynamic Workloads

3.1 What is a Dynamic Workload

A dynamic workload consists of several computational tasks that change the behavior of tasks very frequently over time. Because of its changing behavior, it is hard to predict workload patterns for dynamic workloads. This makes dynamic workload handling a challenging and complex task. In this section, a detailed discussion about the challenges in the dynamic workload and methods to handle a dynamic workload will be discussed in detail.

3.2 Challenges in Dynamic Workload

As discussed earlier dynamic workloads are difficult to manage because of the always changing and evolving behavior. Some of the key challenges of dynamic workloads are given below.

1. Heterogeneity(Zhang & Boutaba 2014)
   Dynamic workloads consist of different types of applications that have their

own and different kinds of resource requirements which is known as the **Heterogeneity**. Heterogeneity can be mainly divided into two parts as

- **Infrastructure heterogeneity and dynamicity**
  With the development of technology machines with different new technologies have been introduced. Hence Data CentersDC has to manage machines with both new and old requirements.
- **Workload heterogeneity and dynamicity**
  Applications with different types of priorities and resource demands have to run in a DC. This heterogeneous environment of the workloads makes it difficult to handle.

2. Cost (Hossain & Song 2016)
   Dynamic workloads have the problem of over-provisioning and under-provisioning. Because of the dynamic nature, it is hard to predict the resource demand hence most of the applications provide resources more than enough to execute the operations. This may lead to a waste of resources. In similar cases, the resources provided may not be enough for the applications to execute which leads to under-provisioning and the failures in application process. This is known as the applications do not meet its **peak demands**. In both cases, part of or the whole cost used for the application process has become a waste.
3. Unpredictability and variability(Cerotti et al. 2012)
   The nature of changing frequently and consists of various types of resources is known as Unpredictability and variability. Both of these aspects lead to an environment that is difficult to handle and make predictions about future works.

3.3 Methods to Handle Dynamic Workload

This section provides a comprehensive study of the different types of methods that we can use to handle a dynamic workload. These methods are mainly categorized into two(2) parts Mathematical/Machine Learning Based Methods and Profiling.

3.3.1 Mathematical/Machine Learning Based Methods

Several mathematical and machine learning-based approaches have been implemented to handle a dynamic workload. The researchers have introduced several ways like using a threshold, using ML approaches, etc. This section provides and detailed description of those methods.

- **Dynamic Resource Allocation using a threshold**(Lin et al. 2011).
  The paper provides a dynamic resource allocation schema that can allocate resources according to the workload changes of the applications. Because of the use of this method applications don't need to allocate resources to meet their peak demands. The algorithm improves resource utilization while minimizing the cost for the users. The dynamic resource allocation schema calculated the workload change of the application and set the interval between two(2) events according to the calculated value. However, since this

is a dynamic workload the workload changes during the application lifetime. So this can cause an unnecessary overhead in application processing. To avoid this issue a threshold is used to decide when to do the resource allocation. By the use of this threshold value, the optimal instance for the resource allocation process will be calculated.

- **Dynamic Workload Handling using Autonomic Computing**(Sah & Joshi 2014).
  Autonomic computing refers to the term that applications can change themselves according to the changes of the application environment without any external help(Without notifying the user). The key features of autonomic computing are **self-monitoring, self-tuning, self-organizing and self-optimizing**. In this method, it takes advantage of autonomic computing to handle a dynamic workload. The created architecture which is based on autonomic computing provides several advantages such as flexibility, cost-effectiveness, dynamicity, and adaptability. In the algorithm, it uses a capacity and utility agent (CUA) and a resource controller.
  The process uses a CUA to collect data about the resources and the capacity of the VMs. According to the data collected from here the initial schedule will be created for the tasks to be executed. Then the process of resource controller will execute. The resource controller consists of three basic functionalities: monitor, analyze, and act. Monitor functionality will detect the changes in the workload and an analyzer will create the appropriate decisions to scale the workload according to the changes. If the analyzer is able to find that the resource requirements could be allocated in a better way then in the act functionality the necessary optimizations will be applied to create an optimized schedule.

- **Multi-Parameter Scheduling Algorithm to Handle Dynamic Workloads** (Hanani et al. 2017)
  The paper proposes a scheduling algorithm that can be used for dynamic workloads. Compared to other methods this algorithm considers multiple parameters to ensure better efficiency. The proposed algorithm **MEOO**(Hanani et al. 2017) consists of three(03) main phases. In the first phase, the primary scheduling was created according to the current workload. Next, the Similarity calculation phase will be executed. In this phase similarity in workloads will be calculated according to the **similarity in size, replication, and message passing**. After calculating the similarity if it is less than a threshold then the scheduling will be improved as the last phase.

- **Enhanced Dynamic Johnson Sequencing Algorithm**(Banerjee et al. 2023)
  The proposed Algorithm named OptiDJS+ uses a combination of ML and RTM techniques to handle the nature of dynamic workloads. OptiDJS+

uses historical data, dynamic resource reconfiguration methods, and necessary adaptions according to the nature of workloads to handle the dynamic behavior of workload and to provide an optimal resource schedule. The use of advanced optimizations to allocate resources intelligently, the use of heuristic methods that allow necessary adaptations according to the workload nature, the use of load balancing is that allows for the distribution of tasks among resources in an efficient manner, and the use of fault tolerance methods to handle failures can be known as the key features of OptiDJS+. As the first step of the algorithm tasks will be prioritized based on the Dynamic Quantum(DQ) value for the task. This DQ value is calculated using the equation **(minimum execution time + maximum execution time)/2**. According to the DQ value, the algorithm will create the execution sequence for the tasks. By using this mechanism OptiDJS+ which is also known as the dynamic Johnson sequencing method has been able to provide high efficiency, lowering operational costs and high scalability while maximizing the resource allocation.

- **A Reinforcement Learning Based Workflow Application Scheduling Approach** (Wei et al. 2018)
  In the cloud environment, there are three(3) types of parties **application users, SaaS providers and IaaS providers**. IaaS providers provide resources for SaaS providers to develop applications and application users will run the applications provided by the SaaS providers. SaaS providers need to find the most suitable and profitable IaaS providers for each task that is going to be developed in the applications. So, the paper introduces a workflow scheduling algorithm using a sequential decision-making approach based on reinforcement learning. The findings of the algorithm can be mapped into the problem of finding the most suitable IaaS providers. The ML approach used in this proposed algorithm is called the Q-learning. Q-learning consists of 2 main phases **initialization phase** and the **decision-making phase**. In the initialization phase, each service provider will be assigned its own execution level. Then in the decision-making phase, the service selection sequence will be created using this calculated execution level. So, by using the created sequence SaaS providers can select the most suitable IaaS provider. Most of the scheduling algorithms depend on the term that the execution time for the tasks can be known in advance. But when it comes to the dynamic workloads that is something hard to predict and the time will change according to the workload changes during the execution. So this method can be used to work with dynamic workloads efficiently.

There are several other methods that use ML and different mathematical techniques to handle the dynamic nature of a workload. (Jayakumar et al. 2020) produce a self-optimized generic workload prediction algorithm based on the ML technique called Long-short Term Memory (LSTM)(Hochreiter & Schmidhuber 1997) to make predictions about the workloads. (Hu et al. 1998) provides an op-

timal load-balancing algorithm for balanced workload distribution among parallel computers. The section 3.3.1 describes another type of well-known method called profiling to handle a dynamic workload.

3.3.2 Profiling

Profiling is a concept used to handle dynamic workloads based on their characteristics. Mainly there are two types of profiling offline profiling and runtime profiling. In offline profiling, it analyzes the performance of the application after its execution. Many systems use this offline profiling information to identify the characteristics of applications. Because of the changes in the network traffic during the run time offline profiling has several limitations such as the difficulties in considering the variations in the application sequence. Runtime profiling can be introduced as a solution to these limitations which tries to capture the characteristics when the application starts, its execution. The captured data will be used to increase application efficiency. CPU time and heap (memory usage) can be known as some of the used runtime profiling information.

- **Dynamic Workload Profiling in Packet Processing Systems**(Wu & Wolf 2008)
  The paper suggests a way to handle task allocation in a packet processing system using runtime profiling. The paper defines a method to partition the application representation into several tasks using profiling information. Task Service time, Edge utilization, and task utilization will be collected as profiling information. Then by using those information tasks will be represented as a workload graph. Finally, by using a task duplication algorithm and a task mapping algorithm, the tasks will be mapped to the processing resources. Since both edge utilization and task utilization are time-dependent this can be known as an instance of the use of profiling methods to handle dynamic workloads.

- **Multi-tier Workload Consolidations in the Cloud**(Ye et al. 2020)
  Multi-tier Workloads are inherently difficult to handle because of their complex structure and various resource demands. The paper provides a method based on empirical profiling to manage the tail latency(duration for the slowest operation to be executed). The first step collect the profiling information and this information will be used to identify similar workloads. Then by using the optimization theory, the tail latency value will be optimized. The created optimization model which is based on the optimization theory will minimize the resource cost while maximizing the workload performance.

- **Profiling for Micro-Services** (Han et al. 2020)
  The dynamic nature of micro-services makes it difficult for micro-services to be placed in the cloud. The paper provides a three(3) step algorithm for micro-services to be placed in the cloud efficiently and flexibly. The novel algorithm is based on profiling and it optimizes the placement of micro-services in Kubernetes clusters. As the first step, a refinement

framework is developed to schedule the placement of micro-services. Then profiling system will be used to track the changes in the workloads and the resource requirements. As the last step, a heuristic algorithm is used for the micro-services placement. Figure:3 shows the design of the refinement framework.

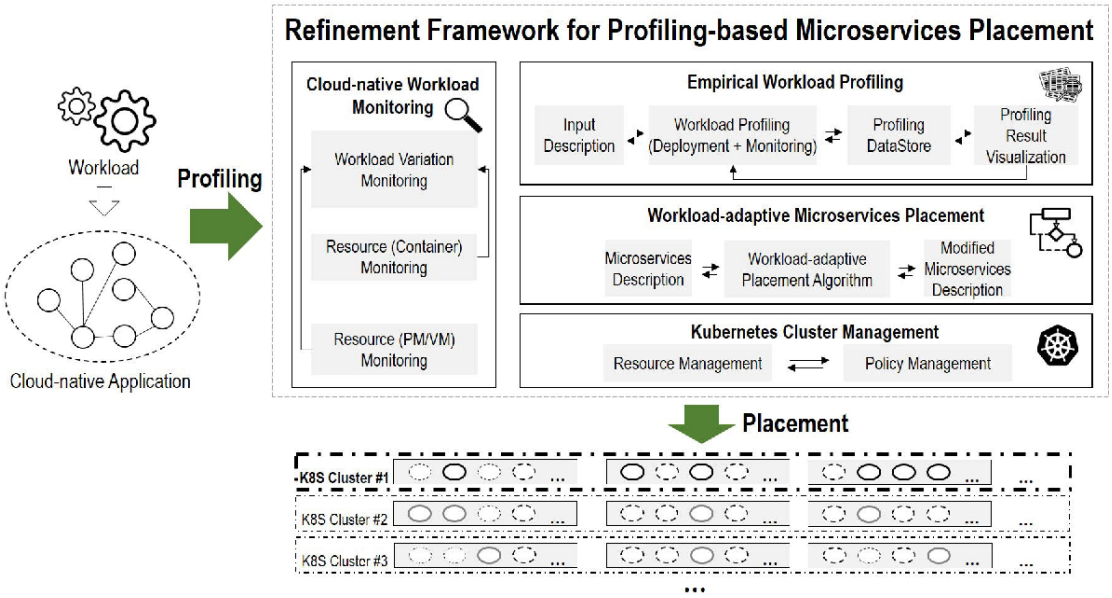

Figure 3: Design of the refinement framework
(Han et al. 2020)

4 Convergence of a Workload

Convergence is an important term that needs to be met for a successful VM migration. Convergence can be discussed according to the two main live migration techniques that is the convergence in Pre-Copy migration and convergence in Post-Copy migration.

- **Convergence in Pre-Copy migration**
  As discussed in section 2.1 during the initial round of Pre-Copy migration all the memory pages will be sent from the source to the destination. For the next round, only the modified pages compared to the previous round will be transferred. This memory translation process continues until there is a point where the number of pages transferred will become a constant value. This frequently editing memory pages set is called the WWS. This point is identified as the **convergence** point and then the algorithm will start the stop and copy phase 1. If the migration process is not able to meet this condition then the Pre-Copy migration algorithm will execute infinitely.

- **Convergence in Post-Copy migration**
  The term convergence isn't directly connected with Post-Copy migration.

Because in the Post-Copy migration, it only transfers the memory pages that need to up the VM at the destination as the first step. Then the rest of the memory will be transferred. If the speed of the memory transferring is not enough then the push-pull(Shribman & Hudzia 2013) method will execute. In this method, the source VM pushes the memory pages while the destination VM quickly pulls the memory pages pushed by the source. Compared with the Pre-Copy migration algorithm that may infinitely run if the transferred memory page count will not become less compared to the previous iteration, the Post-Copy migration algorithm will converge at some point. But if the dirty page rate becomes high the TMT will become high.

This Convergence term also depends on the nature of the workload as static or dynamic. If the workload is static then it is not hard to identify the convergence because the static workload guarantees that its behavior doesn't change during the execution. So for the Pre-Copy migration, the no of pages transferred after the n^{th} iteration will be less than the number of pages transferred during the $(n-1)^{th}$ iteration. But if the workload is dynamic the behavior of the future workload can't be predicted. The dirty page count for the next iteration can be less or more compared to the previous iteration. Because of this nature identifying the convergence of a dynamic workload has become something critical. Table 2 explains the summary of the impact of workload nature on the convergence.

	Pre-Copy Migration	Post-Copy Migration
Static Workload	Since the workload doesn't change there will be a convergence point	Term Convergence does not directly apply here but the page transferring rate has to be handled to reduce the TMT
Dynamic Workload	Since the dirty page rate is up and down frequently it is hard to find a convergence point.	Frequently changing pages may lead to high TMT

Table 2: Convergence according to the nature of the workload.

4.1 Migrating a Dynamic Workload

Migrating a dynamic workload has become challenging because of its frequently changing behaviour which leads to difficulties in identifying the convergence. This section gives a detailed explanation of the existing methods that can used to migrate a dynamic workload.

4.1.1 Adaptive Push-Pull Method for Disseminating a Dynamic Workload

The paper Naik (2022) provides a solution for dynamic workload migration using a technique called **Adaptive Push-Pull**. In CC the CRM will be responsible for managing and distributing the workload across VMs. Figure 4 shows the architecture of the cloud resources. In the concept of Adaptive Push-Pull the push function is used to distribute the workload among the resources and when a VM is overloaded . The pull function is invoked when the VM is underloaded . A combination of these two techniques will be used as the final outcome. The algorithm is based on two concepts called PaP and PoP.

- Push-and-Pull (PaP): Both the push-and-pull functions will be used at the same time and by using a tunable parameter the degree of usage will be calculated for both techniques. According to the workload nature, the parameters can be adjusted to use push, pull, or both the push and pull techniques.
- Push-or-Pull (PoP): In POP both the push and pull will not used at the same time. CRM will select whether to push or pop depending on the workload changes and the VM's load.

VM manager and the CRM can be known as the two main characters used for this process. VM manager is responsible for managing the state within a single resource. If there is an overloaded VM it helps to push the load that is beyond the threshold to an underloaded VM within the same resource. Likewise, if there is an underloaded VM it will pull the load for the underloaded VM from an overloaded VM within the same resource.

CRM is responsible for managing the workload around all the resources. When there is an overloaded VM the CRM will find a VM from another resource that can hold the load beyond the threshold level of the overloaded VM. Then the overloaded VM will migrated. The opposite scenario of this happens when the VM is underloaded. So by using these methods, resources can be managed within the resource as well as outside the resource. This leads to achieving a successful dynamic workload migration.

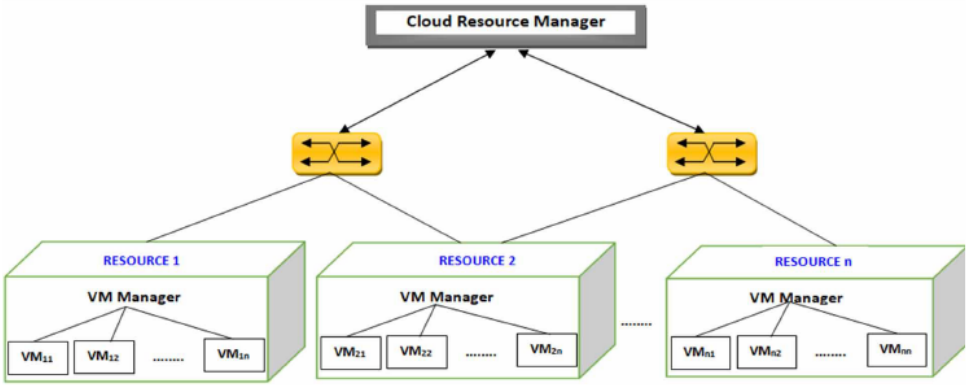

Figure 4: Architecture of cloud resources
(Naik 2022)

4.1.2 Migration Dynamic Multi-tier Aplications

In a multi-tier workload, different layers consist of different resource requirements. Because of this nature dynamic workloads are called inherently dynamic. So the method present in this paper is directly aligned with the methods that can be used to migrate a dynamic workload. The paper provides a method to migrate several VMs concurrently. It suggests two(2) different migration techniques **sequential migration and parallel migration**. In sequential migration, VMs are migrated one after the other while in parallel migration several VMs will be migrated concurrently. Figure 5 depicts the behavior of sequential migration and parallel migration.

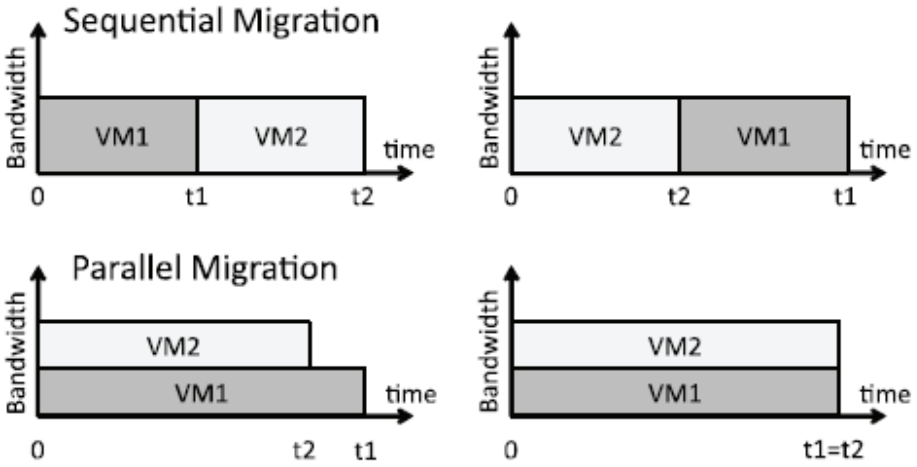

Figure 5: Sequential migration Vs Parallel migration
(Lu et al. 2015)

The algorithm used here is called the **vHaul**(Lu et al. 2015) algorithm. At the first step of the algorithm, VMs will be categorized. The property used for the categorization is that create a separate group from the VMs belonging to the same application. Then for the VMs inside the same group, the product of the resource utilization and migration time will be calculated. According to the value of the product, the VMs will be sorted and the migration order will be created here. To reduce the impact generated by the pending requests VMs with smaller product values will be migrated first.

4.1.3 Gossip Learning Resource Allocation Protocol

Dynamic virtual machine consolidation has to face several difficulties due to the use of static thresholds. Hard to predict future workloads, and difficulties in handling various types of workloads are some of them. The solution provided by the paper is named **GLAP**(Khelghatdoust et al. 2016). Since this method has the ability to predict future workloads it can be used to migrate a dynamic workload as well. The steps involved in GLAP are as follows. GLAP uses continuous monitoring of the resource demands of the VMs and it uses the combination of a gossip-based

learning algorithm and Q-learning to predict the behavior of future workloads. Based on these predicted results the VMs will be migrated without overloading.

5 Critical Analysis of Work

This section gives a comprehensive analysis of how researchers have migrated dynamic workloads, what challenges they have faced, and how they have overcome those issues. Further, a comparison between the methods that can be used for dynamic workload handling is discussed in section 3

5.1 Comparison of Dynamic Workload Migration Techniques

The paper Naik (2022) used a method called **Adaptive Push-Pull** to migrate a dynamic workload. The main challenges that they have faced are,

- Need to distribute the workload across resources in a manner that prevents overloading.
- Need to react to the changes in the dynamic workload.
- Need to reduce the page fault rate in VM migration.

The proposed solution and the benefits that came with the solution are as follows.

- The proposed Adaptive Push-Pull system is used to manage the workload across resources in a proper manner
- The proposed method contains a feature that allows dynamic switching between the push function and the pull function which allows to respond to the changes in the dynamic workload
- CRM and the VM managers are able to manage the load balancing while satisfying a minimal no of page faults.

By using techniques the proposed Adaptive Push-Pull algorithm has successfully overcome the issues in the dynamic workload migration. However, this method is not able to handle the issues related to the multiple workflows and power consumption.

The paper (Lu et al. 2015) provides a migration method for multi-tier workload applications. The challenges that have been addressed in this paper are as follows.

- Since the multi-tier workload applications consist of multiple interrelated layers it is complex to manage all of these together.
- Also when migrating several VMs of multiple layers at the same time, it affects the application performance.

The solution proposed for the above problems is known as **VHaul** the method it used to solve the issue is as below.

- Create an efficient scheduling by considering the resource utilization and the migration time.

The researchers have evaluated the performance of the proposed method in the Xen platform and Apache Olio web application. The results have shown that VHaul has been able to reduce the tail latency by 70%.(Lu et al. 2015). However, for the evaluation, the paper used a web server and a database server, and the

model was tested in the two application platforms called Xen platform and Apache Olio web application. Testing the solution in more applications which consists of different types of environments may lead to identifying further issues and improving the performance of vHaul.

Also, the paper (Khelghatdoust et al. 2016) has provided a way based on Gossip Learning. The main challenges they have solved using the proposed solution are as follows.

- The use of fixed and static thresholds made it difficult to handle the workloads with multiple patterns(Treat all the workloads in the same way)
- Since the future workload is not predicted, it may lead to incorrect decisions.

GLAP has been introduced as the solution to the above challenges. So the solutions are as follows.

- It provides a threshold-free approach. So the problems encountered with the static and fixed threshold would not be there.
- Also since there is no threshold value to handle the workloads it uses a combination of gossip-based learning algorithm and Q-learning. This combination also allows for predicting future workloads.

By using these solutions GLAP has overcome the issues encountered in handling and migrating dynamic workloads. However, the paper also provides some future works to enhance the performance of the GLAP algorithm. Evaluating the workload under bursty workload patterns and using the network topology to reduce energy consumption are some of the future enhancements described in the paper. Table 3 shows the summary of the challenges, solutions, and future enhancements for the discussed dynamic migration methods.

	Challenges	Solutions	Limitations
Adaptive Push-Pull	• Distribute the workload across resources in a manner that prevents overloading • Respond to the changes in the dynamic workload. • Minimize the no of page faults in VM migration.	• Adaptive Push-Pull System • Dynamic switching between the push function • use of CRM and VM managers	• Handles the issues related to the multiple workflows and power consumption.
vHaul	• Difficulty in handling multiple interrelated layers. • Migrating several VMs of multiple layers at the same time affects the application performance.	• Provide an efficient scheduling algorithm by considering the resource utilization and the migration time. migration time	• Testing the solution in more applications that consist of different types of environments.
GLAP	• The Use of fixed and static thresholds made it difficult to handle the workloads with multiple patterns. • Difficulties in predicting the future workloads.	• Provides a threshold-free approach. • Uses a combination of gossip-based learning algorithm and Q-learning for predicting future workloads.	• Evaluating the workload under bursty workload patterns. • Use the network topology to reduce energy consumption

Table 3: Dynamic workload migration techniques

5.2 Comparison of Dynamic Workload Handling Techniques

Several types of techniques have been conducted to handle dynamic workloads. This paper mainly discussed about two(2) types of techniques machine learning/mathematical-based techniques and profiling.
The common challenges that have been described in the papers are as below.

- Difficulties in predicting the behavior of a dynamic workload.
- Need for an optimal method that reduces the cost and provides better methods for resource allocation
- Use of static thresholds may lead to under-provisioning or over-provisioning. (Allocate resources is not enough for the application execution or More than enough resources will be allocated)

(Lin et al. 2011) provides a dynamic resource allocation method with the use of a threshold value to effectively allocate resources when the workload changes. With the use of threshold-based resource allocation schema, the above issues have been solved while providing several advantages such as

- According to the changes in the workload the algorithm will re-allocate the resources.
- Because of that the cost of the work will be reduced and the problem of resource utilization will be solved.
- The implementation of the algorithm is simple which reduces the complexity of the whole process.

However, the proposed algorithm has limited adaptability when the resource requirements are complex.

The paper (Sah & Joshi 2014) defines a method for efficient dynamic workload distribution using the concept called Autonomic computing. Autonomic computing has the advantage of self-organizing and self-monitoring. With the use of those abilities, the proposed algorithm provides several advantages such as,

- flexibility and cost-effectiveness in workload handling.
- Dynamically adapt the behavior according to the changes in the workload
- High adaptability to work with complex CC environments with the need for minimal human actions.

Compared to dynamic threshold-based resource allocation schema this method has high adaptability. However, the solution has not been evaluated for physical machines. So it requires further investigations with different types of workload requirements.

(Hanani et al. 2017) provides a multi-parameter scheduling method for the management of dynamic workloads. Compared to other methods such as (Sah & Joshi 2014) and (Lin et al. 2011) this method provides a more efficient scheduling. The algorithm used for the process is called **MEOO**(Hanani et al. 2017). Its advantages are listed below.

- Provides efficient scheduling by considering multiple parameters.
- Increase performance by reducing the scheduling overhead.

- Results in a better throughput and makespan.
- provides a lower cost for the consumers while saving energy.

The paper suggests some future works such as the use of statistical methods for workload similarity prediction, and the use of queue theory when creating the scheduling.

The paper (Banerjee et al. 2023) uses a ML technique called OptiDJS+. It takes advantage of used optimizations to allocate resources intelligently. The advantages of the (Banerjee et al. 2023) are as follows.

- Compared to the FCFS(Zhao & Stankovic 1989) the OptiDJS+ has better performance, resource utilization and cost-effectiveness.
- provides an efficient load balancing that reduces the number of bottlenecks while ensuring the smoothness of the application process.
- provides an enhanced fault tolerance by using the intelligent job scheduling method.

By the use of these techniques, OptiDJS+ has been able to find a scheduling algorithm that can react to changes in dynamic workloads. However, the advanced techniques used here such as intelligent job scheduling may make the implementation and maintenance of this technique a little bit complex.
The papers (Wu & Wolf 2008),(Ye et al. 2014) and (Han et al. 2020) provide a well-known technique called **profiling** to handle the dynamic workload behavior. The main idea behind the concept used by those papers is to gather runtime profiling information such as CPU time, memory usage, and service time and react according to the changes in the collected information. (Wu & Wolf 2008) uses a duplication and mapping algorithm that is able to capture the changes in the workload by using the collected profiling information. (Ye et al. 2014) used the profiling information to identify similar workloads and then used an optimization model to optimize the tail latency. Since the profiling information helps to identify similar workloads, this method takes advantage of recalculating the resource requirements for similar workloads. (Han et al. 2020) has used a combination of profiling system and refinement framework to identify the placement of micro-services with dynamic workloads. All of these methods have been able to minimize the resource cost compared to the traditional methods while maximizing the performance.

The two(2) types of techniques discussed, machine learning/mathematical-based techniques and profiling have both advantages as well as limitations. When thinking about the term scalability machine learning-based techniques have more scalability to handle complex and and large workloads. Also, it can optimize the performance of an application based on several factors. However, the methods used to manage large workloads are complex, making maintenance difficult and the application failure rate high. Also in machine learning-based methods creating a proper model that does not overfit or underfit is essential and directly affects the application performance. Since there should be a properly trained model. In profiling the concept is easy to understand and implement which provides easy

maintenance. However, the scalability is less compared to the machine learning-based methods. Hence if the workload is highly dynamic, large, or complex the use of machine learning-based methods is preferred.

6 Conclusion and Future Works

The report provides a review of different types of dynamic workload handling methods and some of the already existing dynamic workload migration strategies. It starts by explaining the terms, cloud computing, virtualization, and migration techniques. This provides an in-depth explanation of the nature of dynamic workloads and techniques that can be used to migrate a dynamic workload. Withing the discussed techniques there are two main types Machine learning-based methods and the use of profiling. Finally, concludes by discussing the existing methods for migrating a dynamic workload.
According to the discussed methods the use of machine learning-based methods or profiling depends on the nature of the workload. For highly dynamic or complex workloads, it is recommended to use Machine learning-based methods while profiling is used when easy implementation and maintenance are needed. When considering the advantages and challenges of already existing dynamic workload migration techniques the use of a **hybrid technique** that combines the advantages of both techniques can be introduced as a future work. The hybrid method should be intelligent enough to select the dynamic workload handling technique based on the nature of workload or the invention of a new algorithm that combines both techniques is needed as a future work.

References

Ahmad, R. W., Gani, A., Ab. Hamid, S. H., Shiraz, M., Xia, F. & Madani, S. A. (2015), 'Virtual machine migration in cloud data centers: a review, taxonomy, and open research issues', *The Journal of Supercomputing* **71**, 2473–2515.

Alamdari, J. F. & Zamanifar, K. (2012), A reuse distance based precopy approach to improve live migration of virtual machines, *in* '2012 2nd IEEE International Conference on Parallel, Distributed and Grid Computing', pp. 551–556.

Altahat, M. A., Agarwal, A., Goel, N. & Kozlowski, J. (2020), 'Dynamic hybrid-copy live virtual machine migration: Analysis and comparison', *Procedia Computer Science* **171**, 1459–1468. Third International Conference on Computing and Network Communications (CoCoNet'19).
URL: *https://www.sciencedirect.com/science/article/pii/S1877050920311352*

AWS (2024), 'Amazon web services'.
URL: *https://aws.amazon.com/*

Banerjee, P., Roy, S., Modibbo, U. M., Pandey, S. K., Chaudhary, P., Sinha, A. & Singh, N. K. (2023), 'Optidjs+: A next-generation enhanced dynamic johnson sequencing algorithm for efficient resource scheduling in distributed overloading within cloud computing environment', *Electronics* **12**(19), 4123.

Cerotti, D., Gribaudo, M., Piazzolla, P. & Serazzi, G. (2012), Flexible cpu provisioning in clouds: A new source of performance unpredictability, *in* '2012 Ninth International Conference on Quantitative Evaluation of Systems', IEEE, pp. 230–237.

Choudhary, A., Govil, M. C., Singh, G., Awasthi, L. K., Pilli, E. S. & Kapil, D. (2017), 'A critical survey of live virtual machine migration techniques', *Journal of Cloud Computing* **6**(1), 1–41.

Christopher, C. (2005), Live migration of virtual machines, *in* 'NSDI, 2005'.

Fernando, D., Bagdi, H., Hu, Y., Yang, P., Gopalan, K., Kamhoua, C. & Kwiat, K. (2016), Quick eviction of virtual machines through proactive live snapshots, pp. 99–107.

Fernando, D., Yang, P. & Lu, H. (2020), Sdn-based order-aware live migration of virtual machines, *in* 'IEEE INFOCOM 2020 - IEEE Conference on Computer Communications', pp. 1818–1827.

Galloway, M., Loewen, G. & Vrbsky, S. (2015), Performance metrics of virtual machine live migration, *in* '2015 IEEE 8th International Conference on Cloud Computing', pp. 637–644.

GCP (2024), 'Google cloud platform'.
URL: *https://cloud.google.com/*

Gong, C., Liu, J., Zhang, Q., Chen, H. & Gong, Z. (2010), The characteristics of

cloud computing, *in* '2010 39th International Conference on Parallel Processing Workshops', pp. 275–279.

Han, J., Hong, Y. & Kim, J. (2020), 'Refining microservices placement employing workload profiling over multiple kubernetes clusters', *IEEE access* **8**, 192543–192556.

Hanani, A., Rahmani, A. M. & Sahafi, A. (2017), 'A multi-parameter scheduling method of dynamic workloads for big data calculation in cloud computing', *The Journal of Supercomputing* **73**, 4796–4822.

He, T. (2021), 'Migration management in software-defined networking-enabled edge and cloud computing environments', *degree of Doctor of Philosophy, School of Computing and Information Systems, THE UNIVERSITY OF MELBOURNE, ORCID: 0000-0002-5472-7681* .

Hines, M. R., Deshpande, U. & Gopalan, K. (2009), 'Post-copy live migration of virtual machines', *SIGOPS Oper. Syst. Rev.* **43**(3), 14–26.
URL: *https://doi.org/10.1145/1618525.1618528*

Hochreiter, S. & Schmidhuber, J. (1997), 'Long short-term memory', *Neural computation* **9**(8), 1735–1780.

Hossain, M. A. & Song, B. (2016), 'Efficient resource management for cloud-enabled video surveillance over next generation network', *Mobile Networks and Applications* **21**, 806–821.

Hu, Y., Blake, R. J. & Emerson, D. R. (1998), 'An optimal migration algorithm for dynamic load balancing', *Concurrency: practice and experience* **10**(6), 467–483.

Jayakumar, V. K., Lee, J., Kim, I. K. & Wang, W. (2020), A self-optimized generic workload prediction framework for cloud computing, *in* '2020 IEEE International Parallel and Distributed Processing Symposium (IPDPS)', IEEE, pp. 779–788.

Kashyap, S. & Singh, A. (2023), 'Prediction-based scheduling techniques for cloud data center's workload: a systematic review', *Cluster Computing* **26**, 1–27.

Khelghatdoust, M., Gramoli, V. & Sun, D. (2016), Glap: Distributed dynamic workload consolidation through gossip-based learning, *in* '2016 IEEE International Conference on Cluster Computing (CLUSTER)', IEEE, pp. 80–89.

Lin, W., Wang, J. Z., Liang, C. & Qi, D. (2011), 'A threshold-based dynamic resource allocation scheme for cloud computing', *Procedia Engineering* **23**, 695–703.

Lu, H., Xu, C., Cheng, C., Kompella, R. & Xu, D. (2015), vhaul: Towards optimal scheduling of live multi-vm migration for multi-tier applications, *in* '2015 IEEE 8th International Conference on Cloud Computing', IEEE, pp. 453–460.

Malhotra, L., Agarwal, D., Jaiswal, A. et al. (2014), 'Virtualization in cloud computing', *J. Inform. Tech. Softw. Eng* **4**(2), 1–3.

Microsoft (2024), 'Microsoft azure'.
URL: *https://azure.microsoft.com/en-us*

Miller, R. (2008), 'Failure rates in google data centers', *Data Center Knowledge* .

Mohammed, B. & Kiran, M. (2014), 'Experimental report on setting up a cloud computing environment at the university of bradford', *arXiv preprint arXiv:1412.4582* .

Naik, K. J. (2022), 'An adaptive push-pull for disseminating dynamic workload and virtual machine live migration in cloud computing', *International Journal of Grid and High Performance Computing (IJGHPC)* **14**(1), 1–25.

Nayyer, M. Z., Raza, I. & Hussain, S. A. (2019), Chapter two - revisiting vm performance and optimization challenges for big data, Vol. 114 of *Advances in Computers*, Elsevier, pp. 71–112.
URL: *https://www.sciencedirect.com/science/article/pii/S0065245819300129*

Oracle (2024), 'Oracle — cloud applications and cloud platform'.
URL: *https://www.oracle.com/*

Patel, M., Chaudhary, S. & Garg, S. (2018), 'Improved pre-copy algorithm using statistical prediction and compression model for efficient live memory migration', *International Journal of High Performance Computing and Networking* **11**(1), 55–65.

Red Hat, Inc (2024), 'Redhat virtualization'.
URL: *https://www.redhat.com/en/technologies/virtualization/enterprise-virtualization*

Sah, S. K. & Joshi, S. R. (2014), Scalability of efficient and dynamic workload distribution in autonomic cloud computing, *in* '2014 international conference on issues and challenges in intelligent computing techniques (ICICT)', IEEE, pp. 12–18.

Sahni, S. & Varma, V. (2012), A hybrid approach to live migration of virtual machines, *in* '2012 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)', pp. 1–5.

Shribman, A. & Hudzia, B. (2013), Pre-copy and post-copy vm live migration for memory intensive applications, *in* 'Euro-Par 2012: Parallel Processing Workshops: BDMC, CGWS, HeteroPar, HiBB, OMHI, Paraphrase, PROPER, Resilience, UCHPC, VHPC, Rhodes Islands, Greece, August 27-31, 2012. Revised Selected Papers 18', Springer, pp. 539–547.

SolarWinds Worldwide, LLC (2024), 'Solarwinds virtualization manager'.
URL: *https://www.solarwinds.com/virtualization-manager/use-cases/virtualization-monitoring?CMP=BIZ-RVW-SWTH-mngmt*

Strunk, A. & Dargie, W. (2013), Does live migration of virtual machines cost

energy?, *in* '2013 IEEE 27th International Conference on Advanced Information Networking and Applications (AINA)', pp. 514–521.

V2 Cloud Solutions, Inc (2024), 'V2 cloud'.
URL: *https://v2cloud.com/?referrer=d42d1d2810c5b902e16313e3647f01686XaeD2bHbIO0zE urldate = 2024-01-26*

Wei, Y., Kudenko, D., Liu, S., Pan, L., Wu, L. & Meng, X. (2018), A reinforcement learning based workflow application scheduling approach in dynamic cloud environment, *in* 'Collaborative Computing: Networking, Applications and Worksharing: 13th International Conference, CollaborateCom 2017, Edinburgh, UK, December 11–13, 2017, Proceedings 13', Springer, pp. 120–131.

Wu, Q. & Wolf, T. (2008), Dynamic workload profiling and task allocation in packet processing systems, *in* '2008 International Conference on High Performance Switching and Routing', IEEE, pp. 123–130.

Xing, Y. & Zhan, Y. (2012), Virtualization and cloud computing, *in* 'Future Wireless Networks and Information Systems: Volume 1', Springer, pp. 305–312.

Ye, K., Shen, H., Wang, Y. & Xu, C.-Z. (2020), 'Multi-tier workload consolidations in the cloud: Profiling, modeling and optimization', *IEEE Transactions on Cloud Computing* **10**(2), 899–912.

Ye, K., Wu, Z., Wang, C., Zhou, B. B., Si, W., Jiang, X. & Zomaya, A. Y. (2014), 'Profiling-based workload consolidation and migration in virtualized data centers', *IEEE Transactions on Parallel and Distributed Systems* **26**(3), 878–890.

Zhang, Q. & Boutaba, R. (2014), Dynamic workload management in heterogeneous cloud computing environments, *in* '2014 IEEE Network Operations and Management Symposium (NOMS)', IEEE, pp. 1–7.

Zhao, W. & Stankovic, J. A. (1989), Performance analysis of fcfs and improved fcfs scheduling algorithms for dynamic real-time computer systems, *in* '1989 Real-Time Systems Symposium', IEEE Computer Society, pp. 156–157.

Index