2021 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)

2021 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom) | 978-1-6654-3574-1/21/$31.00 ©2021 IEEE | DOI: 10.1109/ISPA-BDCloud-SocialCom-SustainCom52081.2021.00039

E-VM: An Elastic Virtual Machine Scheduling Algorithm to Minimize the Total Cost of Ownership in a Hybrid Cloud

Milan M. Shetti
Rocket Software
mshetti@rocketsoftware.com

Bingzhe Li
Oklahoma State University
bingzhe.li@okstate.edu

David H.C. Du
University of Minnesota, Twin Cities
du@umn.edu

***Abstract*—A hybrid cloud that combines both public and private clouds is becoming more and more popular due to the advantages of improved security, scalability, and guaranteed SLA (Service-Level Agreement) at a lower cost than a separate private or public cloud. The existing studies rarely consider VM migrations in a hybrid cloud environment with dynamically changed VM workloads. From an enterprise's perspective, these migrations are necessary to minimize the cost of utilizing public clouds and guarantee SLAs of VMs in a hybrid cloud environment. In this paper, we propose an elastic VM allocation and migration algorithm for a hybrid cloud, called E-VM, to fully utilize the resources in a private cloud and to minimize the cost of using a public cloud while guaranteeing the SLAs of all VMs. The E-VM considers the bi-direction migration between private and public clouds. Two components, VM-predictor and VM-selector, are designed and implemented in E-VM to determine if a migration has to be triggered between private and public clouds and which VMs will be migrated to the opposite cloud, respectively. Moreover, E-VM is designed based on the existing public cloud pricing models and can be easily adapted to any cloud service provider. According to simulator results based on a set of captured industrial VM traces/workloads and additional experiments directly on a real-world hybrid cloud, the proposed E-VM can significantly reduce the total cost of using the public cloud compared to the existing VM migration schemes.**

I. Introduction

Cloud computing has become a prevalent platform for offering computing services and storing data for various applications. One of the vital cloud computing features is its scalability and elasticity of both computing and storage resources. Many startups and small companies have used public clouds as their IT infrastructure. Pay-As-Your-Grow (PAYG) basis offers these small companies benefits of saving the cost compared to the traditional software license plus maintenance contract. On the other hand, some large IT companies (e.g., IBM, Hewlett Packard Enterprise (HPE) and Dropbox), who maintain their own cloud environments (i.e., private cloud), have begun to use public clouds and their private cloud (i.e., hybrid cloud) to meet their computing and storage demands. For example, cloud tenants (e.g., IBM, Hewlett Packard Enterprise (HPE) and Dropbox) purchase cloud computing services from cloud providers (e.g., Amazon Web Service (AWS) [1], Google Cloud [2], and Microsoft Azure [3]) to enhance their private cloud capabilities. The public cloud can be used to offer additional computing and storage resources if needed.

A hybrid cloud, a mixture of private and public clouds, can offer both advantages of private and public clouds. First, a hybrid cloud can improve security and privacy by putting sensitive data in the private cloud. Meanwhile, the public cloud component offers elasticity and scalability to meet the service level agreements (SLAs) of users/VMs when the demands of workloads grow up, or a demand spike happens. The research studies related to hybrid cloud can be categorized into two aspects, VM migration and cloud profits. For the VM migration, previous studies [4]–[11] proposed optimization schemes based on different objectives to schedule VMs between different locations or clouds. However, they either statically reassigned VMs to different clouds or ignored the prices of hybrid clouds, which are not sufficient for handling dynamical workloads in hybrid cloud environments. For the cloud profits, they [6], [12], [13] used different pricing strategies to maximize the profits for either individual users or public cloud providers, but are not sufficient for the cloud tenants maintaining a hybrid cloud (more detailed discussion of related work can be found in Section VII). However, currently many issues of hybrid cloud are not thoroughly investigated. For example, how to migrate virtual machines between private and public clouds? And, what is a cost-efficient way to do VM migration?

In this paper, we focus on minimizing the total cost of ownership (TCO) of an enterprise company that uses a hybrid cloud. To simplify the problem, we assume that each VM may contain multiple containers for one independent service. Those enterprises normally own a private cloud with limited resources and also run VM in the public cloud when having the increased or unpredictable bursty demands from VMs such as utilizations of CPU, memory, storage, etc. Thus, some VMs must be dynamically migrated to the public cloud to guarantee the specified SLAs of all services. The public cloud will charge based on the usages of resources by the VMs and obviously the payment for the public cloud may significantly increase the total cost of ownership (TCO). How to minimize the TCO in a hybrid cloud becomes a crucial issue and interesting research topic. To address this problem, three main questions need to be answered: 1) When to trigger a VM migration? 2) Which direction the VM migration should be: from private cloud to public cloud or from public cloud to private cloud? And 3)

Which VMs should be selected for the migration?

To answer all three questions, we propose a scheduling algorithm for hybrid cloud, called E-VM, to minimize the TCO when using a hybrid cloud. The E-VM consists of two major components: VM-predictor and VM-selector. VM-predictor determines when to trigger a migration between private and public clouds and which direction for migration. VM-selector decides which VMs are selected to be migrated. Moreover, the VM-selector takes the public cloud's pricing model into consideration so that the E-VM can optimize the VM migration to achieve a much lower TCO than others.

The organization of the paper is as follows. Section II introduces the background of different types of cloud environments and the pricing models of modern public clouds. Section III formulates the issue that we try to solve and introduces the motivation of this work. The proposed scheme is described in Section IV. The conducted experiments are discussed in Section V and the experimental results in a real system are shown in Section VI. Section VII briefly summarizes the related work. Finally, a conclusion is drawn in Section VIII.

II. Background

A. Public, Private and Hybrid Cloud

Public Cloud: The "pay as you use" consumption model for the public cloud is very attractive to software developers to get a fast start. At the initial stage of development, the required CPU, memory, and network bandwidth as well as the test data are relatively small. So, convenience over cost is a reasonable and favorable trade-off. In this paper, we assume the applications or services developed are based on VMs. Different pricing strategies/models may affect the virtual machine placement strategies. In general, public cloud is a good choice for a) bursting temporary workloads, b) test and development environments which require a fixed set of data to be exported/imported during test and development processes, and c) low to modest amount of storage usage.

Private Cloud: Prior to the public cloud's popularity, enterprises procured best of breed compute, storage and networking from different vendors, with their proprietary set of tools to support applications and manage/integrate those disparate components together. Therefore, it is much more complicated when compared with the experiences of the public cloud. However, the private cloud provides full controllability and improved security for industries to handle their own data.

Hybrid cloud: The hybrid cloud combines the public cloud's benefits with that of the private cloud to provide a) highly scalable, b) cost-effective, and c) high-performance cloud environment with guaranteed SLAs. The most remarkable technology which makes cloud computing feasible is the advent of server virtualization. With server virtualization (i.e., VMs), it is now possible to move VMs between servers in a datacenter or from the private cloud back and forth to the public cloud. In a hybrid cloud, either for cost, performance or control reasons, the need to move VMs between public and private clouds also requires data to be migrated across extremely distributed domains.

B. Pricing Schemes in Cloud Computing

Currently, many public cloud services like Amazon Web Service (AWS) [1], Google Cloud [2], and Microsoft Azure [3] provide several types of cloud computing services including compute, database, analytic, blockchain, etc. The instance types cover from various combinations of CPU, memory, storage, and networking capacities to satisfy requirements from different tenants. By reviewing the pricing models from different cloud providers, we find them sharing similar services and the pricing models can be categorized with two perspectives. One is based on the types of instances, which provide different resource configurations/combinations. For example, general-purpose instances contain balanced CPU, memory, storage, and network resources. The type of memory- or CPU-optimized instances includes more memory or CPU resource than other types. Therefore, they can be used for supporting different applications.

According to the property of "pay as you use" in the public cloud, instances with different pricing plans have various pricing rates. Table I shows the pricing rates of the general-purpose instances with the on-demand plan. Moreover, the users are also charged by transferring their data out (not transferring data in) from the cloud as shown in Table II. Different locations (e.g., Ohio or California) also have different pricing rates. Some cloud providers give a separate storage price rate called Elastic Block Store (EBS) for different types of storage devices associated with an instance as seen in Table I. Our study in this paper is based on a pricing model shown in Table I. We assume that the cloud providers give a pool of resources, and a resource in the pool can be discretely and incrementally added. Moreover, the scheme is independent to the pricing model and can be easily applied to other types of price models.

Finally, the SLAs of all VMs are hard to be fully satisfied due to the possible failures of hardware devices [14]. Therefore, cloud providers always provide an advertised annual or monthly failure rate like 0.00001 percent, which means the total downtime in one month or one year is about 0.00001% (equivalent to about five minutes of downtime per year). If the

TABLE I: Pricing model in Amazon Web Services (AWS) [1]

	vCPU	Memory (GiB)	Cost (per hour)
m5.large	2	8	\$0.096
m5.xlarge	4	16	\$0.192
m5.2xlarge	8	32	\$0.384
m5.4xlarge	16	64	\$0.768
... ...			
General Purpose SSD (gp2)			\$0.1 per GB-month

TABLE II: Pricing of transferring VMs out of clouds in AWS

Amazon [1]		Microsoft [3]		Google [2]	
Size	\$/Month	Size	\$/Month	Size	\$/Month
<1GB	0	<5GB	0	<1TB	0.12
<9.99TB	0.09	<9.99TB	0.087	<10TB	0.11
next 40TB	0.085	next 40TB	0.083	10TB+	0.08
next 100TB	0.07	next 100TB	0.07		
>150TB	0.05	next 350TB	0.05		

TABLE III: SLA violation penalty model [15]

Monthly Uptime Percentage	Penalty rate (service credit)
>99.9%	0
>99%	10%
>95%	25%
<95%	100%

downtime is higher than the advertised value [15], there will be a penalty for the cloud providers (discounts for users) as seen in Table III. In this paper, we follow a similar penalty model as Table III. Since the private cloud has limited resources, it is also possible that some misclassification happens and the management does not migrate some VMs to the public cloud on time. As a result, the utilization of VMs in the private cloud is overflowed (i.e., resources are saturated). So, there will be a penalty when the violations of the SLAs happen. The percentage of downtime is formulated by the total number times of detected resource overflows divided by the total number of times the monitoring and evaluating were done and the penalty cost is proportional to the daily cost of private cloud resource usage.

III. Problem Description

In this section, we describe the problem that we intend to solve. We also present two baseline approaches that will be compared with our approach later. The problem is based on VM management in a hybrid cloud from an enterprise perspective. The enterprise maintains a hybrid cloud including its own private cloud and an outside public cloud. The private cloud has a fixed amount of resources, and the public cloud can always provide adequate resources as the enterprise asked and plans to pay for. A set of long-running virtual machines (VMs) are considered in the hybrid cloud. These VMs support different types of user applications and services. The utilizations of each VM in terms of CPU, memory and storage are dynamically changed since the required resources of different tasks are up and down during different hours and days. The enterprise manages these VMs to satisfy all requirements of VMs as their service-level agreements (SLAs). Due to a large number of VMs used by the enterprise and the demands for resources are dynamically changing, not all VMs can fit in the private cloud, and some VMs must be allocated to the public cloud to ensure the requirements of SLAs. Therefore, to maximize the enterprise's profit, we want to minimize the total cost of using the hybrid cloud while simultaneously satisfying the SLAs of all VMs.

As discussed in Section II, the resources as well as the cost of private cloud including personnel, maintenance and electricity are fixed. So, to minimize the total cost of the hybrid cloud is to minimize the cost of the public cloud. The cost of the public cloud comes from the following three aspects. One is the hourly rate of resources used by VMs in the public cloud. The second is the fee of transferring data from the public cloud out. The last one is the penalty cost ($cost_{penalty}$) if any violations of SLAs happen. Thus, in one period T such as in one month, the total cost can be the summation of these three terms as shown in Eq. (1).

$$cost_{tot} = \sum_{t}^{T} cost_{t,M_{pub}} + p_{tr} \sum_{t} tr\_out_t + cost_{penalty} \quad (1)$$

where $cost_{t,M_{pub}}$ indicates the total resource hourly cost of the public cloud at the t^{th} hour based on the resource utilization of a set of VMs in the public cloud (M_{pub}). P_{tr} is the price rate of transferring data out of the public cloud in a one-month period. tr_out_t refers to the transfer-out amount of data at the t^{th} hour of this month. Based on the numbers of violations and observations, $cost_{penalty}$ is calculated by the total cost of this month multiplied by the penalty rate shown in Table III. Here we assume the cost of the public cloud is measured and charged hourly as we present in Section II.

To solve these problems, those most related studies can be categorized into two types of approaches. One type [16] (**Baseline#1** or **BL#1** for short) is to trigger VM migration from the private cloud to the public cloud when any resource utilization reaches to a predetermined threshold (named forward threshold or f-TH for short) (e.g., 90%). Based on this overflowed resource, the VMs are sorted based on their utilization of this particular resource. After that, the VMs will be put into a migration queue one by one starting from the VM, which uses the least amount of resource to the VM using the largest amount of resource until the total utilization of the overflowed resource of the remaining VMs is smaller than f-TH. Finally, the VMs in the migration queue will be migrated from the private cloud to the public cloud. Moreover, if overflow happens on multiple resources, following similar steps as mentioned above, the VMs are sorted based on each overflowed resource independently. Then, the VMs are put into different migration candidate queues (one per overflowed resource) until their required corresponding resources of the remaining VMs are below the threshold in the private cloud. The VMs that appeared multiple times in these queues have a higher priority to be selected for migration. The rest of the VMs will be selected following a round-robin manner from different queues until the required resources of the remaining VMs in the private cloud are all below the threshold. For the migration direction from the public cloud to the private cloud, it also has a threshold (named backward threshold or b-TH for short) (e.g., 70%), which is smaller than f-TH. The migration will be triggered from the public cloud to the private cloud only when all resource utilization in the private cloud are lower than b-TH. Selecting VM migration candidates follows a similar aforementioned process, which first ranks VMs based on the utilization of different resources independently and selects VMs with a round-robin manner until all required resource utilization of the remaining VMs are below b-TH.

The other type of approaches [4], [13], [17], [18] focuses on optimizing VM allocation. The approach follows the same procedure as in **BL#1** to trigger the migration when any resource utilization reaches a predetermined threshold. As for how to migrate/allocate VMs and minimize TCO, the problem can be formulated to fully utilize the resources in the private cloud. Thus, the issue becomes to maximally fit VMs into the private cloud such that the unused resources in the private cloud are minimal. To solve the issue, we use the scheme in [18] (**Baseline#2** or **BL#2** for short), which summaries all resources by a combined factor (called fitness). We can give all three resources the same weights and then sort the fitness of all VMs. Then, the VMs will be put into the private cloud queue one by one from the VM with the largest fitness factor to the VM with the lowest fitness factor. If any resource is overflowed by allocating the current VM, we just skip the VM until going through all VMs. In the end, we have heuristically maximized the fitness factor in the private cloud and the remaining VMs will be put into the public cloud.

IV. Algorithm Design

We assume that the cost of running VMs in the private cloud is lower and pretty much fixed. Thus, it is important to maximize the usage of the private cloud and reduce the cost of running VMs in the public cloud. The pricing model in the public cloud follows Table I and Table II in Section II-B. We consider three types of resources (i.e., CPU, memory, and storage) needed by each VM in this paper. That is, if a VM is allocated with the required resources of CPU, memory and storage, it will satisfy its SLA. We further assume that there is enough network bandwidth to migrate VMs.

In this section, we start to introduce the proposed E-VM to migrate VMs between private and public clouds. The E-VM consists of two major types of components. One is VM-predictors to determine if any VMs need to be migrated to the other type of cloud. The VM-predictor's purpose is to prevent any resource overflow in the private cloud and reduce costs in the public cloud. The other is VM-selectors that use heuristic algorithms to select a set of VMs to be migrated to ensure that each VM has enough resources to satisfy its SLA.

A. Overall Structure

The major issues of a hybrid cloud are to determine when and which VMs need to be migrated. There are two migration directions: from private to public cloud, and from public to private cloud. Migrations from the private to public clouds is to avoid the resource utilization overflow in the private cloud, resulting in violations of the SLAs of VMs. Migrating a set of selected VMs to the public cloud will release some demand of resources such that the resources of the private cloud are adequate for the remaining VMs. Meanwhile, the public cloud can provide an elastic amount of resources for VMs to satisfy their requirements with some extra cost. Therefore, the migration from private to the public cloud is necessary when the demand for resources by VMs is increased due to bursty type of workloads.

Moreover, Migrating VMs from the public cloud to the private cloud is to minimize the TCO. The cost of running VMs in the public cloud is much higher than that in the private cloud. Thus, it is always necessary to save costs by migrating VMs back to the private cloud if there are enough available

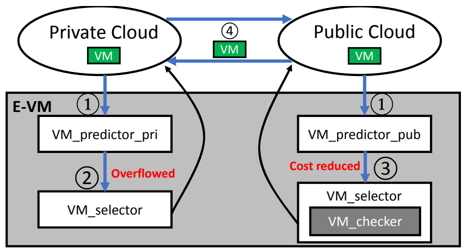

Fig. 1: Overall structure of E-VM.

resources and the resources are underutilized in the private cloud. However, a difficult part is that if the private cloud stays close to the full utilization of its resources, it has a high probability of violating the SLAs of some VMs even with a small burst of demand on resources. The SLA violations will induce some penalty for the enterprise, which increases the TCO as well. On the other hand, if we keep a large margin between available resources and expected used resources in the private cloud, it means that more VMs have to run in the public cloud. Although this is safer to avoid any SLA violations, it also increases the cost when running too many VMs in the public cloud.

As shown in Figure 1, the E-VM follows four major steps for the VM migrations between private and public clouds. ① uses VM-predictors to predict the future utilization of different resources of VMs in either private or public clouds according to the historically collected information. If the overflow of demanded resources is detected, in ② VM-selector is used to select VMs to migrate from private cloud to public cloud to ensure the SLAs of all VMs are satisfied. If a certain cost-reduction condition is satisfied, it will advance to ③. That is, VM-selector will identify a set of VMs to be migrated from the public cloud to the private cloud. The private and public clouds have different criteria to select migration candidates, detailed in the following subsections. The last step, ④, is to do a live VM migration of the selected migration candidates between private and public clouds. In this paper, we assume one of the existing VM live migration techniques can be used [19]–[26] and there is no SLA violation happened according to the live migration. The following subsections provide the details of the E-VM scheme.

B. Migration from Private to Public Cloud

As indicated in Figure 1, there are two steps to decide when and which VMs to migrate. For the resource overflow prediction, the VM-predictor collects and records the utilization of three resources (i.e., CPU, memory, and storage) in the private cloud for each observation period (e.g., every 15 mins). To accurately predict the future resource utilization, a technique called Autoregressive Integrated Moving Average (ARIMA) [27] is used in this paper. ARIMA is widely used for time-series forecasting problems based on non-stationary data. Some other time-series forecasting models such as Long short-term memory (LSTM) [28] can also be used to replace the ARIMA predictor. However, the LSTM model may need

Algorithm 1 VM-selector: migration from private to public

```
1: procedure VM SELECTION - SINGLE FACTOR OVERFLOW
2: // **Assume FA1 is the overflowed factor **//
3:     OF = $max(LP_{mean}, SP_{max}) - private\_FA1$
4:     Reset cur_FA1, cur_FAs, mig_VM and i_prev
5:     for i in VMs do
6:         Comb[i] = coef1*i.FA1 + coef2*i.FA2 + coef3*i.FA3
7:     vm_rank ← ranking Comb with descending order
8:     for i in vm_rank do
9:         if OF > cur_FA1 then
10:            mig_VM.append[i]
11:            cur_FA1 = cur_FA1 + i.FA1
12:            i_prev = i
13:        else
14:            new_FA1 = cur_FA1 + i.FA1 - i_prev.FA1
15:            cur_FAs = $\alpha$*i.FA2 + $\beta$*i.FA3
16:            if OF ¡ new_FA1 & i_prev.FAs ¿ cur_FAs then
17:                mig_VM.remove(i_prev)
18:                mig_VM.append(i)
19:                i_prev = i
20:                cur_FA1 = new_FA1
21:            else
22:                mig_cand.clear()
```

more datasets and take a longer time to predict. Thus, we decide to use the ARIMA model as the predictor.

The VM-predictor-pri predicts the overall utilizations of three resources for the near future (next observation period) in the private cloud. If the demand for any resources is likely to be overflowed, VM migration must be triggered to avoid any SLA violation. Since the demand for resources may be bursty and unpredictable, the ARIMA model's prediction may not be accurate. We use two methods to mitigate the inaccurate prediction. We add a Confidence Interval (CI) to the ARIMA model and use the upper bound as the predicted value. By doing so, the upper bound gives us a margin to tolerate some bursty workloads. Moreover, we use VM-predictor-pri to predict two different numbers of future points. One is a shorter length with few near future points (e.g., next four future points), and the other is a longer length (e.g., sixty future points). According to the ARIMA model, the shorter period can give more accurate results. The longer period is used to predict the trend of resource utilization. So, we use the maximum value of the following two predicted values to determine whether the migration should be triggered or not (i.e., whether the demand for any resource may be overflowed or not): one is the predicted maximum value by shorter period (SP_{max}) and the other predicted average value of the longer period (LP_{mean}). If any of these two values are higher than the resource capability in the private cloud, the migration will be triggered and the process advances to ② in Figure 1.

At ②, the VM-selector selects an appropriate set of VMs for migrating to the public cloud. Since the multiple resources are considered, the VM-selector should find the VMs with high utilization of the overflowed resource(s) but with low utilization of the other resources. If only a single resource is overflowed, we call **single-factor overflow**. Otherwise, called **combined-factor overflow**. The algorithms for these two scenarios are similar but have a little difference. We first describe how Algorithm 1 handles the single-factor overflow. The overflow value (OF) is computed by the predicted value ($max(SP_{max}, LP_{mean})$) minus the resource capacity ($private\_FA1$) of the overflowed resource $FA1$ in the private cloud. If the overflow happens (i.e., $OF > 0$), we use a $Comb$ value to determine which VM is supposed to be migrated. The $Comb$ value is computed for each VM based on its weighted resource utilizations (Line 5). If $FA1$ is the overflowed resource, we use $coef1 = 10$ and $coef2 = coe3 = 1$ to distinguish the VMs with the same $FA1$ via the other factors. We then ranked the VMs based on their $Comb$ values with a descending order (Line 7). After that, the VMs based on the ranked $Comb$ values are put into a migration pool (mig_VM) one by one until the accumulated utilization (cur_FA1) of the VMs in mig_VM is larger than OF.

Moreover, to keep other resources (non-overflowed) in mig_VM as low as possible, we try to replace some VMs to decrease overall utilization in mig_VM. A parameter (cur_FAs) indicates how much utilization of the other resources will be migrated. If $OF < FA1$, we start to check if there are new VMs that have smaller demands for $FA2$ and $FA3$ than the last inserted VM in the mig_VM. The weights α and β reflect the cost overhead of the other two resources in the public cloud. For example, in the public cloud, if only 5% margin for $FA2$ advances to the next price level with a cost increase of \$0.2/h and 10% margin for $FA3$ with an extra \$0.3/h cost, then $\alpha = 0.1$ for $FA2$ and $\beta = 0.3$ for $FA3$. The coefficients can help distinguish the cost influence of different resources in the public cloud.

Similarly, for combined-factor overflow, the VM-selector should pick up the VMs with high utilization on all these overflowed resources and ensure the utilization of non-overflowed resources is as low as possible. First, we need to make sure that the OFs of all overflowed resources are smaller than that in the mig_VM. In other words, after migration, there is no overflow for any factor/resource in the private cloud. The selection scheme prefers to pick up the VMs having high utilization on those overflowed resources (use $Comb$ value in Line 6). This is because some VMs have strong correlations between different resources. For example, one VM with high CPU utilization may also have high memory utilization. So, choosing such a VM to migrate can reduce the utilization of multiple overflowed resources. We use a similar algorithm to replace the VMs with high utilization of non-overflowed resources in the mig_VM with VMs having low utilization of non-overflowed resources.

C. Migration from Public to Private Cloud

To reduce the overall cost, we should consider the VM migration from public cloud to private cloud if there are available resources in private cloud. However, this needs to be done carefully since there is a higher possibility of violating the SLAs of some VMs if the resources in the private cloud are almost fully utilized. Moreover, we do not want to migrate some VMs between public and private clouds back and forth in a short duration. Such a ping-pong migration may cause a high charge since transferring data out from the public cloud

Algorithm 2 VM-selector: migration from public to private

```
1: procedure VM SELECTION
2:     Reset cur_cpu, cur_mem, cur_sto, mig_VM and i_prev
3:     Obtain available private cloud resource margins: $mgn_{cpu}$, $mgn_{mem}$, $mgn_{sto}$
4:     $pub_{cpu}$, $pub_{mem}$, $pub_{sto}$ = pub_level()
5:     for i in VMs do
6:         Comb[i] = coef1*i.CPU + coef2*i.Mem + coef3*i.Storage
7:     vm_rank ← ranking Comb with descending order
8:     if $mgn_{cpu} > pub_{cpu}$ and $mgn_{mem} > pub_{mem}$ then
9:         for  i in vm_rank do
10:            if i not in mig_VM and $mgn_{cpu}$ > cur_cpu + i.cpu and $mgn_{mem}$ > cur_mem + i.mem and $mgn_{sto}$ > cur_sto + i.sto then
11:                mig_VM.append[i]
12:                cur_cpu = cur_cpu + i.cpu
13:                cur_mem = cur_mem + i.mem
14:                cur_sto = cur_sto + i.sto
15:    if $pub_{cpu}$ > sum(i.cpu in mig_VM) and $pub_{mem}$ > sum(i.mem in mig_VM) then
16:        mig_VM.clear()
17:    else
18:        for i in mig_VM do
19:            if $cost_{mig}(i) > cost_{normal}(i)$ then
20:                mig_VM.remove(i)
21: end
22: procedure PUB_LEVEL()
23:    cpu_util = max($LP_{mean}.cpu$, $SP_{max}.cpu$) - level_cpu
24:    mem_util = max($LP_{mean}.mem$, $SP_{max}.mem$) - level_mem
25:    return $pub_{cpu}$, $pub_{mem}$
```

is not free as seen in Table II. Similar to the migration from private cloud to public cloud, there are two steps to follow, but with different conditions and a different selection strategy.

To avoid the ping-pong migration, the algorithm needs to satisfy two conditions to trigger the migration from public cloud to private cloud. We use the same predictor (named VM-predictor-pub in the public cloud) and prediction model to estimate the future resource utilization of the public cloud. One (**Cond#1**) is that the migration can make the charge of public cloud usage reduced by at least one pricing level according to the public cloud pricing model. Since the pricing model, as shown in Table VI, is discrete, migrating some VMs back to the private cloud may not reduce the overall cost. Moreover, the reduced cost should be larger than the transferring out cost. The other condition (**Cond#2**) is that the private cloud has enough unused resources to accept the migrated VMs. The upper bound of the confidence interval (CI) is used for predicting the utilization of resources. We take CI = 90% as a default value in all evaluation experiments.

The VM-selection process is shown in Algorithm 2. First, the E-VM predicts how much resource utilization should be reduced in the public cloud such that the pricing rate can be degraded to a lower level (Lines 22-25). Then, based on the predicated resource utilization of public and private clouds, we check if the current situation satisfies the **Cond#2** (Line 8). After that, we put the VMs into vm_rank based on their $Comb$ values. Similar to Algorithm 1, $coef$ values are computed from pub_FA/Phy_FA (Phy_FA is the resource capacity of private cloud for the factor FA). Starting from the highest $Comb$ value in vm_rank, we put VMs in the mig_VM pool just before violating the condition (**Cond#2**).

TABLE IV: Private cloud resource configurations for the scenarios with different numbers of VMs

# of VMs	vCPU	Mem(GB)	Storage (TB)
40	16	32	1
100	32	96	16
400	128	512	64

Moreover, the **Cond#1** (Lines 15-16) checks if the cost is reduced based on the VMs in the migration pool. If not, the migration pool will be cleared and no VMs will be migrated to the private cloud. In **Cond#1**, the transferring out fee is considered (Lines 18-22) to determine if the overall cost is reduce or not.

D. E-VM Overhead Discussion

The overheads of the E-VM scheme are mainly from extra space and computing time. One is the space overhead from recording VM resource utilization information for both private and public clouds. The information includes the history of the overall utilization of CPU, memory, and storage in the private and public clouds. Assume we use the most recent 25 hours (100 historical monitored points) to predict future one hour and four hours (future four and sixty points) and data are stored with float format (4 bytes). Therefore, total amount of history information is 100 * 3 (resources) * 2 (clouds) bytes = 600 bytes. For each VM, we record their most recent 4 points. Then, we have 12*N bytes (N is the total number of VMs). So, the overhead of recorded information is really small (400 VMs only need the space overhead of about 5.5KB).

The other overhead is the time required from building and executing a time-series prediction model. We investigated the overhead in a system with Intel(R) Xeon(R) CPU E5-2620 v3 2.4GHz processors. We use the Python3 with the statsmodels library. The execution time of building and executing an ARIMA prediction model is about 0.107s. Compared to the monitor time interval (15 minutes in this work), the execution time is tolerable.

V. Experimental Results

A. Environment Description

We captured a mixture workload with 16 applications running on 400 VMs (i.e., 400 individual services) in a hybrid system. The applications are described in Table V. The peak utilizations of CPU, memory, and storage of each VM is captured for every 15 minutes. The lengths of those traces are 1250 hours. We then run simulations based on different migration schemes and public cloud pricing models to calculate their overall cost ($). To investigate the scalability of this work, three scenarios with different numbers of VMs (40 VMs, 100 VMs and 400 VMs) are considered in these experiments. The scenarios of 40 VMs and 100 VMs select VMs following a round-robin manner from 16 applications. Correspondingly, we set up several private cloud configurations for these three scenarios, as shown in Table IV. The pricing model follows Table II, Table III and Table VI for migration cost, penalty cost and normal cost, respectively.

TABLE V: Configurations of virtual machine workloads with running different applications

Applications	VM Type	IO Pattern	# of VMs	Avg. vCPU	Avg. memory (GB)	Avg. storage (GB)
CephS3	Mission Critical	Sequential	30	0.08	0.12	62.85
MySQL_Prod	Mission Critical	Random	30	0.98	7.76	219.06
MySQL_Test	Test and Dev	Random	50	1.76	4.53	36.28
Exchange_Tx	Mission Critical	Mixed	30	0.68	0.78	495.89
Intranet_ProjTx	Business Critical	Sequential	40	0.73	0.83	171.38
Internet_ProjTx	Business Critical	Random	30	0.50	1.27	167.93
CouchDB	Staging	Random	20	0.55	1.61	47.30
VDI	Business Critical	Mixed	30	0.34	0.44	36.90
VDI_Europe	Business Critical	Sequential	30	0.48	0.07	202.96
Bak_NonTradApp	Backup	Streaming	20	0.52	1.69	60.50
VDI_Asia	Business Critical	Random	20	0.32	0.05	243.53
Bak_TradApp	Backup	Sequential	30	0.28	0.10	204.42
SAP_Ariba_Prod	Mission Critical	Random	10	0.90	1.52	2157.16
SAP_Ariba_Stage	Staging	Sequential	10	0.08	0.12	25.35
SAP_Ariba_Test	Test and Dev	Random	10	0.64	2.00	16.02
HDFS_Output	Test and Dev	Sequential	10	0.24	0.02	47.24

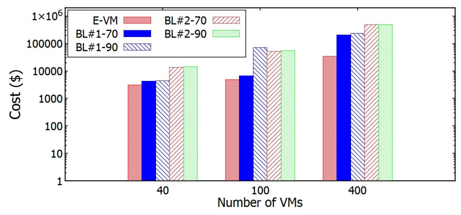

Fig. 2: Overall cost comparison with different numbers of VMs.

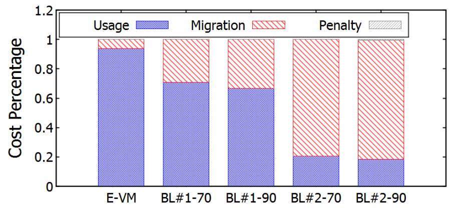

Fig. 3: Percentage of different costs with 40 VMs workload.

B. Overall Comparison

In this subsection, we make comparisons between the proposed scheme and the two baselines described in Section III. Baseline1 (BL#1) is based on independent resources to select VMs as migration candidates. Baseline2 (BL#2) uses a heuristic algorithm to optimize VM locations when the migration is triggered. By default, the trigger migration threshold is set to either 70% or 90% for these two baselines, denoted by BL#1-70, BL#2-70, BL#1-90, and BL#2-90, respectively. As shown in Figure 2, we can find that the proposed scheme can reduce the overall cost by 1.4x - 4.6x, 1.4x - 11.6x, and 6.1x - 14.6x for the workloads of 40 VMs, 100 VMs, and 400 VMs, respectively. To analyze the results, we breakdown the cost into three categories (*Usage*, *Migration* and *Penalty*). *Usage* refers to the cost of normal utilization of VMs running in the public cloud. *Migration* refers to the migration cost when egressing out from the public cloud to the private cloud. *Penalty* demonstrates the penalty cost when SLA violations happen in the private cloud. The percentage costs of these three categories are indicated in Figure 3. We can find that the proposed E-VM achieves more than 90% normal usage cost, and the cost of combining migration and penalty is less than 10%. BL#1 and BL#2 get much larger migration cost (25%+ and 75%+, respectively) than E-VM. The reasons are threefold: 1) The static threshold cannot provide an accurate prediction of utilization trends and thus it potentially causes a higher migration overhead between private and public clouds. 2) The combined resource consideration in E-VM takes care of the potential correlations between different resources (e.g., CPU and memory) and correctly selects VMs to fit in either private or public clouds. It fully utilizes the private cloud resource and also potentially reduces the number of migrations triggered. 3) The migration checker filters out some VM migration candidates by comparing the cost between staying in public and egressing out from the public cloud to the private cloud. As a result, it prevents ping-pong migrations between private and public clouds, thus reducing the overall cost.

TABLE VI: Pricing models of Amazon [1], Microsoft [3] and Google [2]

	Unit	vCPU	Memory (GiB)	Cost ($/h)	Storage ($/GB-month)
Amazon	m5.large	2	8	0.096	0.1
Microsoft	n1-std	1	2	0.036	0.075
Google	Av2	1	3.75	0.0475	0.17

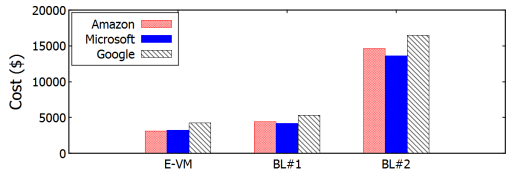

Fig. 4: Overall costs by using different public clouds with 40 VMs workload.

C. Different Cloud Pricing Models

In this subsection, we investigate the overall cost of these schemes when using different cloud platforms. As shown in Table VI, we pick three similar general-purpose instances from Amazon Web Service, Microsoft Azure, and Google Cloud as our resource units. Following the discussion in Section II-B, the resources in Table VI are basic units, and the total amount of resources in the public cloud can be scaled based on

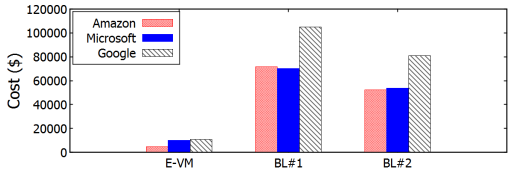

Fig. 5: Overall costs by using different public clouds with 100 VMs workload.

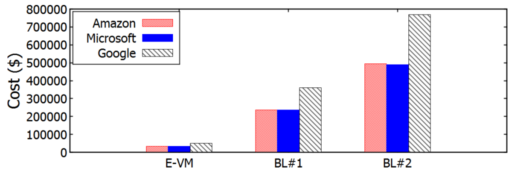

Fig. 6: Overall costs by using different public clouds with 400 VMs workload.

how many resources VMs will use. The pricing models of transferring VMs out of clouds are following Table II.

The total cost of different platforms with running three different workloads (40 VMs, 100 VMs and 400 VMs) are shown in Figure 4, Figure 5 and Figure 6 respectively. First, the proposed E-VM scheme always gets the least cost compared to the two baselines no matter what platforms we are using. The reasons are described in the previous subsection. To horizontally compare different platforms based on the pricing model in Table VI, Google obviously obtains the highest cost among three cloud platforms with running 40, 100, and 400 VMs for all three schemes. The reason is that Google has a much higher storage price rate (1.7x - 2.3x higher) and a higher cost rate of transferring data out (12% higher under 10TB) than those from Amazon and Microsoft. To compare between Amazon and Microsoft, they achieve similar cost. For some cases like the workload of 40 VMs, Amazon is higher than Microsoft, while Amazon obtains a lower cost in 100 VMs for E-VM and BL#2. To figure out a deeper understanding, we can normalize the cost to one vCPU and one GiB memory cost. Therefore, Amazon has \$0.048/vCPU and \$0.012/GiB memory, while Microsoft has \$0.036/vCPU and \$0.018/GiB memory. Obviously, memory-intensive workloads are better to use Microsoft or Amazon platforms, while CPU-intensive workloads prefer to use the Microsoft cloud platform.

Lesson#1*, people can always find a similar pricing model in different cloud providers. However, the little difference may give users a significant benefit based on the requirements of workloads.*

D. Individual Types of Applications

In this subsection, we run each type of applications individually in the hybrid cloud. As mentioned in Section V-A, 16 types of workloads have different properties. We run them separately to see their total cost effect. The private cloud configuration follows the 40-VM configuration shown in Table IV. The price model follows the Amazon pricing model in Table VI.

As we see in Figure 7, in total 16 workloads, on average, the proposed E-VM achieves 36.3% and 80.7% lower cost than BL#1 and BL#2, respectively. Moreover, it outperforms the two baselines with nine workloads. E-VM and BL#1 achieve similar in four workloads (e.g., Exchange_Tx and SAP_Ariba_stage). BL#2 always obtains the highest cost due to a high migration overhead. Meanwhile, BL#1 has a lower cost than E-VM in three workloads (i.e., CephS3, MySQL_Prod, and SAP_Ariba_Test). The reason is that these three workloads have more frequent dramatic resource utilization changes and thus the predictor in E-VM cannot accurately predict the trend of utilization changes. As a result, a larger penalty is induced due to more resource overflows in E-VM and thus, incurs a higher cost than BL#1. For the mixed workloads discussed in Section V-B, dramatic resource utilization changes of few workloads might be canceled or mitigated by mixing different workloads. Therefore, for those workloads of 40-, 100-, and 400 VMs, E-VM always outperforms the two baselines.

E. Effect of Separate Features

In this subsection, we discuss the effect of different individual features. Starting from BL#1, we add one feature each time until it becomes E-VM as described below:

- **BL#1:** is the baseline scheme which uses static threshold to trigger migration and consider each utilization separately.
- **+Dynamic:** adds dynamic utilization prediction based on BL#1.
- **++Combined:** considers correlation between factors/resources based on +Dynamic.
- **+++Mig:** uses a migration checker in the public cloud based on ++Combined, which is **E-VM**.

We compare the above four schemes with three workloads. The pricing model is the same as that in Section V-B. To conveniently see the cost reduction, the total cost of BL#1 is normalized to 1. From Figure 8, Figure 9, and Figure 10, we can find that in general the total cost is gradually decreased from BL#1 to +++Mig (E-VM). On average, +Dynamic obtains 19% lower cost than BL#1, ++Combined gets 16% lower cost than +Dynamic and +++Mig achieves 59% lower cost than ++Combined. There are some exception in Figure 8 and Figure 9. For example, +Dynamic gets a higher cost than BL#1. The reason is similar to the scenario of CephS3 in Section V-D. Due to irregular changes of resource utilization, the utilization predictor inaccurately predicts the future utilization and triggers some unnecessary migrations between public cloud and private cloud. For another exception in Figure 9, when considering correlations between resources, we can migrate VM more efficiently and fully utilize private cloud resources, resulting in less migration overhead and less normal usage in the public cloud. However, when any resource utilization faces a burst, it induces a higher risk of violating SLAs in the private cloud. Therefore, the SLA violation penalty in this case is higher than the others.

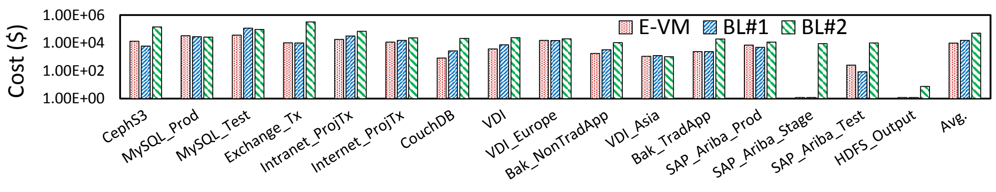

Fig. 7: Total migration size and number of migrated VMs for two schemes.

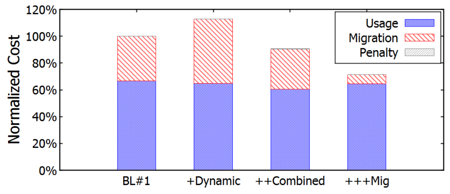

Fig. 8: Total cost of individual schemes with the 40 VM workload.

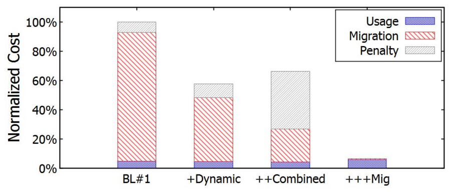

Fig. 9: Total cost of individual schemes with the 100 VM workload.

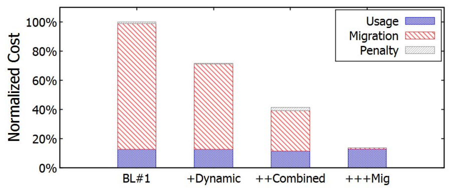

Fig. 10: Total cost of individual schemes with the 400 VM workload.

***Lesson#2**, based on the modern cloud pricings, the price strategy of cloud providers is to discourage users from migrating VMs out from public clouds.*

VI. Real System Evaluation

The real system uses 1000 Virtual Machines with an average of 8 virtual CPUs per VM. Initially 500 Virtual Machines were in the private cloud and 500 Virtual Machines were hosted in Amazon elastic services. The workload used was the clone copies of the workload of an IT company. This workload allows us to get the performance of a realistic workload as close as possible. The experimental environment had 100 HPE DL380 Gen 10 Servers. Each server configured with Intel Xeon Model, Gold 6248R Processor with 24 core 3.0GHz processors, 35.75 MB L3 Cache, 2 @10.4 GT/s UPI and 2933 MT/s DDR4 memory 1 TB per socket. Each Server is configured to host 4 Virtual Machines using VMware vSphere 6.7 U3. The Network Controller on each server, 10/25Gb 2-port LFR-SFP28 MCX4121A-ACFT Adapter. The Storage Controller on each server, P408i-a w/2GB cache 8 port modular Smart Array with 24 SFF SSDs in Front and 6 SFF SSDs in the rear.

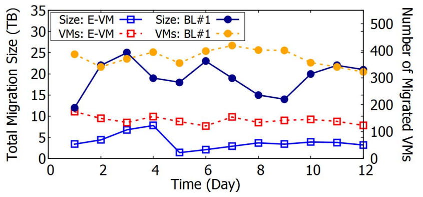

Fig. 11: Total migration size and number of migrated VMs for two schemes.

According to the simulation results in Figure 6, BL#2 obtains the worst performance and may cause much expensive cost in the real system. So, only the proposed E-VM and BL#1 are applied in a real system. We made a comparison for these two schemes. We investigate the data movement in hybrid clouds. The less data is moved, the better it is from reducing the cost point of view because the data in and out is one of the important cost parameters in the public cloud. As shown in Figure 11, it is clear to see that the BL#1 model performs worse than the proposed scheme. Specifically, BL#1 migrates 4.9x more data (2.6x more VMs) than the proposed E-VM scheme. This is primarily because the simple decision model trying to keep a large number of VMs in the private cloud makes the algorithm aggressively move VMs from the public cloud to the private cloud. Also, this BL#1 algorithm more aggressively picks up VMs of low CPU utilization from the public cloud to migrate.

VII. Related Work

For the hybrid cloud's research topics, people mainly focused on two aspects of the hybrid cloud. In the first aspect, the previous studies [4]–[13], [29] mainly focused on task/VM allocation. For example, Liang *et al.* [4] proposed a cost-driven model for scheduling tasks/VMs in hybrid clouds by using a firework algorithm based algorithm. VM allocation is only considered by the availability of resources in an environment with small-scaled physical machines. Finally, they minimized the cost from a user's perspective. H2-D2 [12] is an approach focusing on the problem of multi-objective VM reassignment for large and hybrid data centers. Multiple dimensions of the infrastructure optimization including the overhead of running IT infrastructure, reliability and migration were independently considered. However, they statically reassigned VMs to different clouds, which are not sufficient for handling dynamical workloads. Also, they did not consider the pricing model for a hybrid cloud environment. The other aspect is that different roles in hybrid cloud set different objectives. For example, Liu *et al.* [13] proposed a scheme to decide the resources in a private cloud and schedule the requests from users to private or public clouds. Also, they tried to decide the optimal prices for the public cloud service providers by using a Stackelberg game model. Therefore, this work is trying to solve the issues for public cloud service providers. From the perspective of cloud providers, they try to set up proper pricing strategies for their products [1], [3], [13].

In summary, most of the above work did not manage VM allocation and migration in a hybrid cloud from an enterprise point of view. They did not fully consider the dynamically changing workloads of VMs in hybrid clouds. They in general ignore the utilization correlations between different resources and lack a comprehensive consideration of pricing models.

VIII. Conclusion

In this paper, we propose a smart migration scheme for hybrid clouds, called E-VM, to target on fully utilizing the resources in the private cloud and minimizing the cost of using the public cloud. The E-VM considers bi-directional migrations between private and public clouds. Two components (VM-predictor and VM-selector) are built in the E-VM to determine if there is any need for migration between private and public clouds and which VMs will be migrated to the other cloud. Moreover, the E-VM is designed based on one pricing model in Amazon Web Services and can be extended to different pricing models from other cloud providers. According to the experimental results, the proposed E-VM can reduce the total cost of using the public cloud by 1.4x - 14.6x compared to the existing VM migration schemes.

IX. Acknowledgement

This work was partially supported by NSF I/UCRC Center Research in Intelligent Storage and the following NSF awards 1439622, and 1812537.

References

[1] Amazon web service. https://aws.amazon.com/.

[2] Google cloud. https://cloud.google.com/.

[3] Microsoft azure. https://azure.microsoft.com/en-us/.

[4] Helan Liang, Yanhua Du, Enting Gao, and Jinghan Sun. Cost-driven scheduling of service processes in hybrid cloud with vm deployment and interval-based charging. *Future Generation Computer Systems*, 107:351–367, 2020.

[5] Jyotiska Nath Khasnabish, Mohammad Firoj Mithani, and Shrisha Rao. Tier-centric resource allocation in multi-tier cloud systems. *IEEE Transactions on Cloud Computing*, 5(3):576–589, 2015.

[6] Jinjin Wang, Yonglong Zhang, Junwu Zhu, and Yi Jiang. A double auction vm migration approach. In *2nd EAI International Conference on Robotic Sensor Networks*, pages 141–147. Springer, 2020.

[7] Hua He, Yu Zhao, and Shanchen Pang. Stochastic modeling and performance analysis of energy-aware cloud data center based on dynamic scalable stochastic petri net. *Computing and Informatics*, 39(1-2):28–50, 2020.

[8] Gaochao Xu, Junjie Pang, and Xiaodong Fu. A load balancing model based on cloud partitioning for the public cloud. *Tsinghua Science and Technology*, 18(1):34–39, 2013.

[9] Christopher Clark, Keir Fraser, Steven Hand, Jacob Gorm Hansen, Eric Jul, Christian Limpach, Ian Pratt, and Andrew Warfield. Live migration of virtual machines. In *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation-Volume 2*, pages 273–286, 2005.

[10] Nikos Tziritas, Thanasis Loukopoulos, Samee Khan, Cheng-Zhong Xu, and Albert Zomaya. Online live vm migration algorithms to minimize total migration time and downtime. In *2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 406–417. IEEE, 2019.

[11] Naga Malleswari Tyj and G Vadivu. Adaptive deduplication of virtual machine images using akka stream to accelerate live migration process in cloud environment. *Journal of Cloud Computing*, 8(1):3, 2019.

[12] Takfarinas Saber, James Thorburn, Liam Murphy, and Anthony Ventresque. Vm reassignment in hybrid clouds for large decentralised companies: A multi-objective challenge. *Future Generation Computer Systems*, 79:751–764, 2018.

[13] Zhe Liu, Changle Li, Weijie Wu, and Riheng Jia. A hierarchical approach for resource allocation in hybrid cloud environments. *Wireless Networks*, 24(5):1491–1508, 2018.

[14] Jason Lango. Toward software-defined slas. *Communications of the ACM*, 57(1):54–60, 2014.

[15] Oracle paas and iaas public cloud services . https://www.oracle.com/assets/paas-iaas-pub-cld-srvs-pillar-4021422.pdf.

[16] Rahul Ghosh, Giribabu V Paramkusham, Aaron J Quirk, and Upendra Sharma. Selecting virtual machines to be migrated to public cloud during cloud bursting based on resource usage and scaling policies, March 28 2017. US Patent 9,606,826.

[17] Moustafa Najm and Venkatesh Tamarapalli. Vm migration for profit maximization in federated cloud data centers. In *2020 International Conference on COMmunication Systems & NETworkS (COMSNETS)*, pages 882–884. IEEE, 2020.

[18] Srinivas Byatarayanapura Venkataswamy, Indrajit Mandal, and Seetharam Keshavarao. Chicwhale optimization algorithm for the vm migration in cloud computing platform. *Evolutionary Intelligence*, 13(4):725–739, 2020.

[19] Michael Nelson, Beng-Hong Lim, Greg Hutchins, et al. Fast transparent migration for virtual machines. In *USENIX Annual technical conference, general track*, pages 391–394, 2005.

[20] Michael R Hines, Umesh Deshpande, and Kartik Gopalan. Post-copy live migration of virtual machines. *ACM SIGOPS operating systems review*, 43(3):14–26, 2009.

[21] Takahiro Hirofuchi, Hidemoto Nakada, Satoshi Itoh, and Satoshi Sekiguchi. Enabling instantaneous relocation of virtual machines with a lightweight vmm extension. In *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pages 73–83. IEEE, 2010.

[22] Hai Jin, Li Deng, Song Wu, Xuanhua Shi, and Xiaodong Pan. Live virtual machine migration with adaptive, memory compression. In *2009 IEEE International Conference on Cluster Computing and Workshops*, pages 1–10. IEEE, 2009.

[23] Jihun Kim, Dongju Chae, Jangwoo Kim, and Jong Kim. Guide-copy: fast and silent migration of virtual machine for datacenters. In *SC'13: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, pages 1–12. IEEE, 2013.

[24] Fei Zhang, Guangming Liu, Xiaoming Fu, and Ramin Yahyapour. A survey on virtual machine migration: Challenges, techniques, and open issues. *IEEE Communications Surveys & Tutorials*, 20(2):1206–1243, 2018.

[25] Xiang Zhang, Zhigang Huo, Jie Ma, and Dan Meng. Exploiting data deduplication to accelerate live virtual machine migration. In *2010 IEEE international conference on cluster computing*, pages 88–96. IEEE, 2010.

[26] TianZhang He, Adel N Toosi, and Rajkumar Buyya. Performance evaluation of live virtual machine migration in sdn-enabled cloud data centers. *Journal of Parallel and Distributed Computing*, 131:55–68, 2019.

[27] George EP Box and David A Pierce. Distribution of residual autocorrelations in autoregressive-integrated moving average time series models. *Journal of the American statistical Association*, 65(332):1509–1526, 1970.

[28] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[29] Keke Gai and Meikang Qiu. Reinforcement learning-based content-centric services in mobile sensing. *IEEE Network*, 32(4):34–39, 2018.