Geo-Distributed Multi-Tier Workload Migration Over Multi-Timescale Electricity Markets

Sourav Kanti Addya, *Senior Member, IEEE*, Anurag Satpathy, *Graduate Student Member, IEEE*, Bishakh Chandra Ghosh, *Student Member, IEEE*, Sandip Chakraborty, *Member, IEEE*, Soumya K. Ghosh, *Senior Member, IEEE*, and Sajal K. Das, *Fellow, IEEE*

***Abstract*—Virtual machine (VM) migration enables cloud service providers (CSPs) to balance workload, perform zero-downtime maintenance, and reduce applications' power consumption and response time. Migrating a VM consumes energy at the source, destination, and backbone networks, i.e., intermediate routers and switches, especially in a Geo-distributed setting. In this context, we propose a VM migration model called Low Energy Application Workload Migration (*LEAWM*) aimed at reducing the per-bit migration cost in migrating VMs over Geo-distributed clouds. With a Geo-distributed cloud connected through multiple Internet Service Providers (ISPs), we develop an approach to find out the migration path across ISPs leading to the most feasible destination. For this, we use the variation in the electricity price at the ISPs to decide the migration paths. However, reduced power consumption at the expense of higher migration time is intolerable for real-time applications. As finding an optimal relocation is $\mathcal{NP}$-Hard, we propose an *Ant Colony Optimization* (ACO) based bi-objective optimization technique to strike a balance between migration delay and migration power. A thorough simulation analysis of the proposed approach shows that the proposed model can reduce the migration time by 25%–30% and electricity cost by approximately 25% compared to the baseline.**

***Index Terms*—Workload migration, migration delay, migration power, ant-colony optimization, multi-tier applications.**

I. Introduction

The advent of geo-distributed cloud service applications has compelled cloud service providers (CSP) to deploy interconnected data centers (DCs), called *Internet Data Centers* (IDC), at multiple geographic locations depending on the user demands. Electricity is proven to be one of the major day-to-day operational costs for running an IDC, so the CSPs typically try to minimize electricity costs by optimizing the usage of DC hosts and other physical infrastructure such as the networks. However, the scenario becomes complicated under the multi-timescale electricity market where the cost of the electricity changes over time [1], and the CSP has connectivity with different electric service providers. For example, Fig. 1 depicts the hourly variation in the electricity price for four providers, namely CAISO, MISO, PJM, and ERCOT in the USA (as of 25th August 2016) [2]. There is a considerable variation in the electricity cost at different times of the day, and it also varies across various electric service providers. Such a market attracts competition among the providers to deliver cheap electricity and calls for a challenging real-time cost optimization problem for the CSPs in deciding the operational infrastructure such that the overall running cost can be minimized.

Existing approaches for optimizing energy cost of CSPs [1], [3], [4], [5] primarily addressed this issue using the following macro-level energy management strategies: (a) service placement and consolidation [1], (b) infrastructure-level operations [3], [6], and (c) energy procurement [4], [5]. However, none of these existing works have considered the micro-management of energy usage during typical data center operations such as service migrations. For example, consolidation of services as discussed in [1] requires migrating the services from one DC to another. However, a DC maintains redundant network connectivity through different Internet Service Providers (ISP) for a reliable, available, and fail-safe system. Therefore, multiple

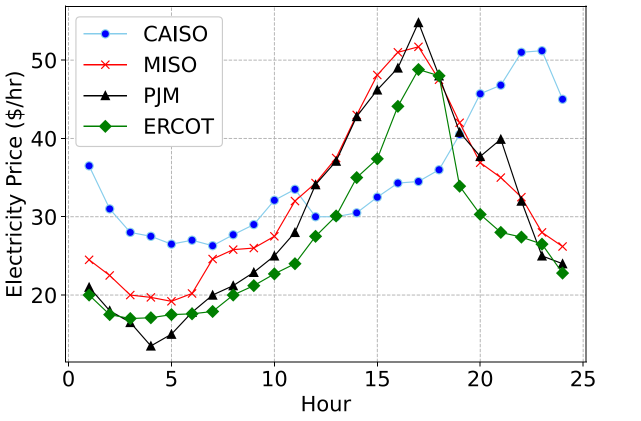

Fig. 1. Hourly variation in the electricity price ($/hr) for four different providers, namely CAISO, MISO, PJM, and ERCOT in the USA [2].

Manuscript received 27 November 2020; revised 20 March 2022; accepted 17 April 2023. Date of publication 27 April 2023; date of current version 8 October 2023. The work of Sajal K. Das was supported in part by the NSF under Grants OAC-2104078 and OAC-1725755. Recommended for acceptance by E. Damiani. *(Corresponding author: Sourav Kanti Addya.)*

Sourav Kanti Addya is with the Department of Computer Science and Engineering, National Institute of Technology Karnataka Surathkal, Mangaluru 575025, India (e-mail: kanti.sourav@gmail.com).

Bishakh Chandra Ghosh, Sandip Chakraborty, and Soumya K. Ghosh are with the Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, Kharagpur 721302, India (e-mail: ghoshbishakh@gmail.com; sandipchkraborty@gmail.com; skg@iitkgp.ac.in).

Anurag Satpathy is with the Department of Computer Science and Engineering, National Institute of Technology, Rourkela, Rourkela 769008, India (e-mail: anurag.satpathy@gmail.com).

Sajal K. Das is with the Department of Computer Science, Missouri University of Science and Technology, Rolla, MO 65409 USA (e-mail: sdas@mst.edu).

Digital Object Identifier 10.1109/TSC.2023.3270921

migration paths exist between a source host and the target host for service migration. Different paths for migrations will incur different energy costs in a multi-timescale energy marketplace [2] both at the backbone ISP and the target DC. Indeed, the power consumption by the ISP networking elements associated with a DC is a significant concern for the CSPs for optimizing the operational costs [7]. In [8], Heller et al. have shown that the networking elements consume 10%–20% of the total energy at the DCs. Considering the large number of service migrations that typically occur per hour over a commercial geo-distributed DC [9], an important aspect would be to reduce the energy per bit vis-à-vis the electricity cost for service migration.

A straightforward solution to the above micro-management problem would be to dynamically decide the migration path to minimize energy consumed per bit of data migration. This is a challenging problem because of the following reasons. (i.) Nowadays, the typical cloud services are multi-tier in nature [10], where a set of virtual machines (VM) are allocated to a running service. A service migration involves the migration of all the associated VMs; however, a single target host may not meet the service level agreement (SLA) for all the associated VMs. Therefore, multi-tier services migration involves a complex decision problem to decide the target hosts and the associated migration paths. (ii.) The optimization problem is bi-objective – we need to reduce the per-bit energy consumption for migration and maximize the migration bandwidth. The migration becomes fast with minimal application downtime. (iii.) Although the decision problem is complex, the solution needs quick convergence as it needs to be taken in real-time.

In this paper, we focus on the micro-optimization of operational cost for service migration across geo-distributed cloud DCs. Our approach meticulously selects the ISP paths that lead to the fast migration of multi-tier cloud services while minimizing energy consumption and reducing electricity costs. We model this as a multi-objective combinatorial optimization problem, which is $\mathcal{NP}$-Hard. Therefore, we develop a fast heuristic-based approach to solve the optimization using *Ant Colony Optimization* (ACO). Combining the optimization problem and the corresponding fast solution approach, we propose **L***ow* **E***nergy* **A***pplication* **W***orkload* **M***igration* (*LEAWM*) in this paper that can find out the optimal ISP paths for multi-tier service migration over a geo-distributed cloud while maximizing the migration bandwidth and minimizing the energy cost. We have implemented *LEAWM* over *CloudSim* simulator, and a thorough evaluation of the system shows that *LEAWM* can reduce the migration time as well as the energy consumption cost due to migration of a multi-tier cloud service.

A preliminary version of this paper was published in [11], where we presented a framework to optimize the electricity cost and time for migrating multi-tier services in a geo-distributed cloud. The major contributions of the current work are as follows:

- We propose a framework called as *LEAWM* that aims to schedule the migration of multiple VMs corresponding to multi-tier applications. We focus on appropriately selecting ISP paths that can reduce the migration time and reduce the per-bit energy consumption in the process (Sections III and IV).
- We propose an efficient migration strategy as a part of *LEAWM* that can efficiently determine the paths for conducting large-scale VM migrations considering the metrics mentioned above (Sections V and VI). Finding migration paths is challenging, especially in multi-timescale electricity markets; we use an ACO-based meta-heuristic approach to solve the above problem (Section VII).
- We perform thorough evaluations and performance comparisons with a baseline to ascertain the proficiency of *LEAWM*. Further, the simulation results confirm the fact that *LEAWM* can reduce the migration time by 25%–30% and electricity cost by approximately 25% in comparison to the baseline (Section VIII).

II. Related Work

Various research works in the literature have proposed methods to optimize the total migration time and overall power consumption due to migration. Both pre-copy and post-copy migration approaches have been adopted as a migration strategy for conducting migrations on a multi-cloud architecture for multi-tier applications.

A. Optimizing Migration Time

Mandal et al. [16] have proposed an intelligent strategy for accurately determining the bandwidth and pre-copy rounds to optimize the migration in a multi-cloud setup. In [19], the authors have proposed *serial* and *parallel* migration techniques for migrating a sequence of VMs for multi-tier applications. Serial migration follows a sequential ordering of VMs, where each VM gets the full share of the LAN bandwidth during migration. Therefore, the migration time for individual VM is reduced. On the other hand, the VMs share the available LAN bandwidth in the case of a parallel migration strategy. In contrast to serial migration, parallel migration reduces the downtime for a multi-tier application. In [17] is introduced an improved serial migration strategy for migrating multiple VMs using a hybrid pre-copy and post-copy approach. The post-copy method allows the VMs to stop their execution at any point in time and copy their boot data to resume execution at the destination. Although this technique can reduce the migration time by minimizing the amount of data to be copied, it often suffers from page faults as the VM's entire memory is not copied. This can lead to delayed response and negatively impact the user experience and satisfaction.

To overcome this drawback, the authors in [20] built a writable working set (WWS) by iteratively scanning the page table and sending it with the VM states to reduce the number of page faults, which consequently improves the service quality. However, the scope of these existing approaches is limited to scheduling independent VMs. On the contrary, modern cloud applications are represented as virtual data centers (VDCs) that comprise multiple interdependent VMs. VDC management often requires relocation of the entire logical application space, which triggers large-scale migrations of dependent VMs [22]. A parallel migration strategy is adopted in [15] for relocating multiple correlated VMs and reducing overall service downtime. Higher migration times often have harmful impacts on the performance

TABLE I
SELECTIVE REVIEW FOCUSED ON VM MIGRATION

Work	Migration Strategy Used (Live)			Parameters			Environment		Migration Type	
	Pre Copy	Post Copy	Hybrid Copy	Migration	Downtime	Power Consumption	LAN	WAN	Single	Concurrent
LEAWM	✓	✗	✗	✓	✓	✓	✗	✓	✗	✓
He et al. [12] (2021)	✓	✗	✗	✓	✓	✗	✗	✓	✗	✓
Addya et al. [13] (2021)	✓	✗	✗	✓	✓	✗	✗	✓	✗	✓
Tziritas et al. [14](2019)	✓	✗	✗	✓	✓	✗	✓	✗	✓	✗
Sun et al. [15] (2018)	✓	✗	✗	✓	✓	✗	✗	✓	✗	✓
Mandal et al. [16] (2017)	✓	✗	✗	✓	✓	✗	✗	✓	✗	✓
Sun et al. [17] (2016)	✗	✗	✓	✓	✓	✗	✓	✓	✗	✓
Jin et al. [18] (2014)	✓	✗	✗	✓	✓	✗	✓	✓	✓	✓
Callegati and Cerroni [19] (2013)	✓	✗	✗	✓	✓	✗	✓	✗	✗	✓
Sahni and Varma [20] (2012)	✗	✓	✗	✓	✓	✗	✓	✗	✓	✗
Hines and Gopalan [21] (2009)	✗	✓	✗	✓	✓	✗	✓	✗	✗	✓

of responsive applications in terms of delayed response. To minimize the total migration time, a *memory compression based VM migration* (MECOM) is proposed in [18], which aims to reduce the amount of memory to be copied to provide fast and stable migration. Some other works that addressed the issue of bandwidth determination, termination conditions, and scheduling the migration of multiple VMs are discussed in [12], [13], [14].

B. Optimizing Power Consumption

Recent literature on minimizing the power consumption of DCs by efficiently relocating VMs has gained much impetus. Due to regional demand differences, transmission inefficiencies, and generation diversity, electricity prices exhibit temporal and geographic variation. Qureshi et al. [23] have characterized the change due to fluctuating electricity prices and argued that the existing distributed systems should exploit such variations for significant economic gains. Gupta et al. [24] have utilized the spatial and temporal variation in electricity price to reduce the operating cost associated with energy consumption. Rao et al. [25] have studied the issue of minimizing the total electricity cost under multiple electricity markets environments while ensuring the application QoS. On the other hand, the authors in [26] have discussed a novel approach for cost savings using geographical load balancing without violating the SLA. Notably, Heller et al. [8] have shown that the networking elements are one of the major consumers of electricity. Since VM migration is a network-intensive task, it leads to increased power consumption of the intermediate networking elements. Frequent migrations can increase electricity consumption at the intermediate nodes and decrease the CSPs' revenue. Hence, there is a need to design a migration strategy for geo-distributed clouds, exploiting the variation in electricity pricing at the intermediate CSPs to reduce the migration overhead. However, reduced networking cost at the expense of higher migration time is futile [27].

Table I provides a comprehensive summary of the literature that we reviewed on VM migration. The table indicates that the existing literature is limited to addressing migration time and power consumption independently. However, the two parameters have a strong correlation, especially while conducting massive-scale migrations over a geo-distributed infrastructure. In this paper, we propose *LEAWM* to jointly reduce the electricity cost due to migration without jeopardizing the migration time, as detailed in the following.

III. Problem Definition

In this paper, we address the problem of scheduling the migration of VMs of multi-tier applications. We consider a scenario where multiple applications are deployed over servers in the form of multiple interconnected VMs. As discussed earlier, this is challenging as (i) migrating a multi-tier service involves finding suitable destination servers for each VM, (ii) identifying the migration paths over the interconnection network such that the per-bit energy overhead is reduced, and (iii) scheduling the migration of such VMs such that the service downtime is minimum.

Specifically, given a geo-distributed physical data center network (or substrate network) comprising multiple data centers at different locations, interconnected through multiple ISPs, we develop an efficient VM migration scheduling mechanism by suitably selecting a migration path that optimizes the per-bit-energy consumption and service downtime over a multi-timescale electricity market. Note that we focus on a single cloud with DCs located at different geographical locations and procuring energy services from providers with cost variations at a temporal scale.

A. System Model

In this section, we provide a detailed discussion of different components of the proposed *LEAWM* framework.

1) Multi-Tier Application Instance: We assume a set of N multi-tier application instances $\mathbb{M} = \{M_1, M_2, \ldots, M_N\}$ where M_n refers the nth multi-tier application instance. Each multi-tier application instance comprises a number of VMs denoted by $\mathbb{V}_n = \{V_1, V_2, \ldots, V_M\}$, where V_m is the mth VM, and M is the total number of VMs within $\mathbb{V}_n$. Typically a VM, $V_m \in \mathbb{V}_n$, requests CPU and memory resources denoted by V_m^{CPU} and V_m^{Mem} respectively [13]. Further, for each VM, we define a location constraint vector α_m that captures the allowable DCs where a VM can be hosted. The variables in the vector $\alpha_m \in \{0, 1\}$ and are assigned values as per (1).

$$\alpha_{m,j} = \begin{cases} 1 : \text{if } V_m \text{ can be placed in DC } j \\ 0 : \text{otherwise} \end{cases} . \tag{1}$$

2) Substrate Network Model: The interconnected DC network is modeled as an undirected weighted graph $\mathbb{G}^S = (\mathbb{N}^S, \mathbb{E}^S)$. Here, $\mathbb{N}^S = \mathbb{N}^{DC} \cup \mathbb{N}^I$, where $\mathbb{N}^{DC}$ denotes the set of substrate networks, and $\mathbb{N}^I$ indicates the set of intermediate ISP's routers/switches in the entire network. We also assume that the graph is connected, i.e., there is at least one route from one DC to any other DC. Every DC $dc_j \in \mathbb{N}^{DC}$ has K servers at its disposal represented by $\mathbb{H}_j = \{h_1, h_2, \ldots, h_K\}$ and uniquely indexed by H_k. Each server $h_j^k \in \mathbb{H}_j$ has resources along two dimensions, i.e., CPU and memory resources captured as $h_j^{k.CPU}$ and $h_j^{k.Mem}$ respectively. The physical links connecting the networking elements are represented by $\mathbb{E}^S = \mathbb{E}^P \cup \mathbb{E}^B$. Here, $\mathbb{E}^P$ indicates the number of intra-DC physical links, and $\mathbb{E}^B$ represents the backbone WAN links. We only consider the transport layer devices (routers, Internet Exchange Points, etc.) of the ISP as the major components in the network, which consume power during their operations. Note that we only consider the migration overhead for inter-DC links in this approach. The backbone links are $\mathbb{E}^B = \{e_1, e_2, \ldots, e_L\}$, where each physical link $e_l \in \mathbb{E}^B$ connects two networking elements (gateways of two interconnecting ISPs) and has the following properties: (i) $c(e_l)$ represents the maximum capacity in terms of bandwidth, (ii) $r(e_l)$ indicates the current availability which implies the availability of resources at a specific time-instance (initialized at the total capacity), and (iii) $d(e_l)$ captures the maximum propagation delay experienced.

B. Multi-Tier Migration Request

In a typical cloud environment, the applications are migrated from one DC to another due to multiple reasons such as route maintenance, handling excess demands, load balancing, etc. All the VMs associated with the application may need to be migrated for a multi-tier application. We define a mapping function Γ_M as per (2) to capture the initial assignment of the VMs over the host servers across the DCs. $\mathbb{H}$ denotes the set of all servers across DCs.

$$\Gamma_M : \mathbb{V}_n \rightarrow \mathbb{H}. \tag{2}$$

For instance, the mapping of a VM v_m can be captured in accordance with (3).

$$\Gamma_M(V_m) = h_j^k \qquad \forall m \in [1, M];\ \forall k \in [1, K];\ \forall j \in [1, |\mathbb{N}^{DC}|]. \tag{3}$$

Given the initial mapping function Γ_M, the VMs of the multi-tier applications often get relocated. In this paper, for a multi-tier application n, we define, $\mathbb{V}'_n \subseteq \mathbb{V}_n$ as the set of VMs that require relocation. The relocation is a complex process involving finding a suitable destination server for each migrating VM. Moreover, as we discussed earlier, VM migration is a network-intensive task and consumes the networking elements' resources, leading to increased energy consumption and reduced revenue. Hence, the overall service migration problem is subdivided into two sub-problems: (i) Determine a feasible destination server and (ii) Determine the migration path, which not only reduces the migration overheads but also minimizes the per-bit energy consumption. Next, we discuss each step in detail.

1) Virtual Machine Remapping: The remapping process addresses the first aspect of the problem by selecting a feasible destination server for the migrating VM. Although there are efficient techniques to address the issue of VM placement [28], [29], in this work, we use a first-fit allocation strategy for VM relocation as the primary aim of this paper is to find optimal relocation paths that optimize both the migration time and the migration cost in terms of energy costs. The remapping function is captured as per (4), where $\mathbb{H}' \subseteq \mathbb{H}$.

$$\Gamma_R : \mathbb{V}_n \rightarrow \mathbb{H}'. \tag{4}$$

The remapping function for a VM V_m is captured as per (5), and is feasible *iff* the constraints expressed in **C1**, **C2**, **C3**, and **C4**, are satisfied. A VM can perform its normal operations if assigned the requisite resources. The Constraints **C1** and **C2** reflect that the relocating server h_j^k satisfies the CPU and memory demands of the migrating VM V_m. On the other hand, **C3** states that remapping is feasible if the target server belongs to an allowable data center for the VM. Finally, **C4** denotes the feasible set of values those the variables can take.

$$\Gamma_R(V_m) = h_j^k, \tag{5}$$

subject to the conditions **C1**: $V_m^{CPU} \leq h_j^{k.CPU}$; **C2**: $V_m^{Mem} \leq h_j^{k.Mem}$; **C3**: $\alpha_{m,j} = 1$; and **C4**: $\forall m \in [1, M];\ \forall k \in [1, K];\ \forall j \in [1, |\mathbb{N}^{DC}|$

2) Finding Remapping Paths: The next step involves finding a suitable migration path that is able to optimize two parameters – *(i.) the total migration time*, and *(ii.) the energy costs incurred during the migration process*. Ideally, finding a path with the minimum migration time involves selecting the path with the highest residual bandwidth. Note that we do not have any dedicated migration paths between two DCs; we use the same in-band Internet path used for communication between the VMs. Selecting the path with the highest residual bandwidth may not always lead to reduced per-bit energy consumption. Moreover, the migration path encompasses traversing the networking elements of multiple ISPs acquiring electricity from disparate service providers experiencing temporal variation in the electricity costs. This emphasizes that the path selection is bi-objective and involves striking a trade-off between the migration time and per-bit energy consumption. The path remapping is captured using the one-to-one mapping function as expressed in (6). It reflects the overall working of the path mapping function, which implies finding a migration path from the source server to the remapped server. Here, $\mathbb{P}^S$ denotes the set of loop-free paths over the interconnection network.

$$\Gamma_P = (\Gamma_M(V_m), \Gamma_R(V_m)) \rightarrow \mathbb{P}^S. \tag{6}$$

Precisely, for a given VM V_m, a migration path selection is not trivial. This is especially challenging when we have multiple geo-distributed DCs having multiple paths connecting any two substrate servers. For instance let $P_m \in \mathbb{P}$ be the selected migration path for migrating V_m connecting $\Gamma_M(V_m)$ and $\Gamma_R(V_m)$. The migration bandwidth $\mathcal{R}_m$ for the path can be derived as per the (7). Note that all the migrations for a given multi-tier application are scheduled over this path irrespective of the strategy. The overall migration bandwidth of the path is dictated

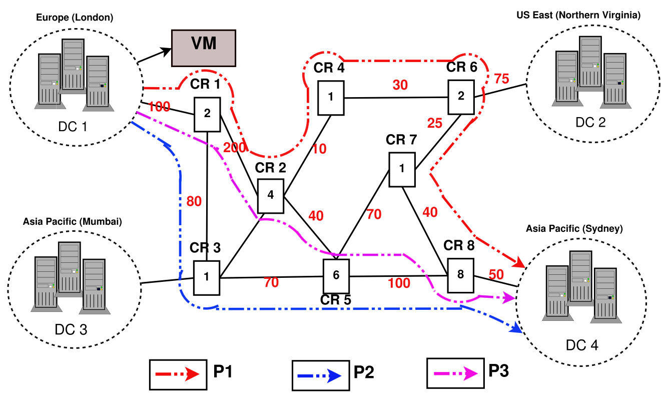

Fig. 2. Interconnection network comprising four DCs, i.e., $DC1$, $DC2$, $DC3$ and $DC4$ connected using the backbone network. $P1$, $P2$ and $P3$ represent three different migration paths for migrating a VM from $DC1$ to $DC4$ over the interconnected network.

by the physical link having the least available capacity.

$$\mathcal{R}_m = \min_{\forall e_l \in P_m} \{r(e_l)\}. \tag{7}$$

To impose restrictions on maximum tolerable migration time, we also impose an additional constraint as reflected by (8). This constraint caps the maximum migration delay under the maximum tolerable delay δ_n for a multi-tier application n. $\mathbb{T}_m^{mig}$ denote the migration delay incurred by V_m. This is done to make the implementation realistic and applicable to real-time and deadline-sensitive applications.

$$\sum_{\forall V_m \in \mathbb{V}_n} \mathbb{T}_m^{mig} \leq \delta_n. \tag{8}$$

In the following subsection, using an example scenario, we highlight the complexity in the decision-making process considering both migration and energy concerns while migrating VMs of multi-tier applications across a Geo-distributed infrastructure.

C. An Example Scenario

To make the example scenario closer to a real-world setting, we consider a geo-distributed infrastructure of interconnected IDCs similar to that of Amazon web services (AWS) [30] and is pictorially depicted in Fig. 2. Although AWS has deployed over 26 IDCs, we restrict the example scenario to 4 interconnected DCs assumed to be located in Europe (London), US-East (Northern Virginia), Asia-Pacific (Mumbai), and Asia-Pacific (Sydney), hereafter referred to as $DC1$, $DC2$, $DC3$ and $DC4$, respectively. The interconnection network, as illustrated in Fig. 2 may not exactly resemble the one used by AWS but is realistic enough to understand the undertaken problem. Further, we also assume the presence of multiple ISPs forming the backbone network to ensure a *reliable*, *available*, and *fail-safe* cloud-delivery system [31].

As a consequence, every IDC has multiple interconnection paths with other IDCs. The intermediate ISPs have routers (CRs) to route the traffic, and these routers receive electricity from different providers and incur additional costs. For simplicity, let us assume a scenario where a multi-tier application only requests a single VM relocation. Let us assume the VM's initial mapping is at $DC1$, and we need to find a suitable destination DC to host the migrating VM. Although the migration path comprises both inter and intra-DC paths, we reduce the inter-DC latency. This is because it is a scarce and costly resource. We further assume that only $DC4$ can host the VM and satisfies the location constraint. As can be observed from the figure, multiple paths connect $DC1$ and $DC4$. Let us assume that VM has 200 *bits* of data to be transmitted, and the electricity cost for sending each *bit* at each CR is written inside the square.

Note that these values are used for this example scenario only. The mentioned value on the edge connecting two devices denotes the transmission capacity of each link. Let us assume the migration path is $DC1 \rightarrow CR1 \rightarrow CR2 \rightarrow CR5 \rightarrow CR8 \rightarrow DC4$. The total cost incurred in migrating 200 bits over this path is 4,000 *units*, and the total time for transmitting all the *bits* is 5 *units*. Alternatively, there is another path $DC1 \rightarrow CR1 \rightarrow CR2 \rightarrow CR4 \rightarrow CR6 \rightarrow CR7 \rightarrow CR8 \rightarrow DC4$. The total cost incurred in migrating 200 *bits* over this path is 3,600 *units* whereas the total transmission time is 10 *units*. Contrary to the above paths, there exists another path $DC1 \rightarrow CR1 \rightarrow CR3 \rightarrow CR5 \rightarrow CR8 \rightarrow DC4$, which can reduce both the cost to 3,400 and the transmission time to 4 *units* respectively. Hence, finding the most optimal path that optimizes both the migration overhead and cost of using intermediate devices (specifically routers) is not trivial and demands us to find a suitable technique that optimizes both.

IV. *LEAWM* Architecture: Joint Optimization of Migration Delay and Migration Power

As we discussed earlier, the *LEAWM* architecture deals in building an efficient migration plan for already-assigned VMs of multi-tier applications allocated across a Geo-distributed infrastructure. These relocations can be triggered due to various activities such as maintenance, load balancing, disaster recovery, etc., undertaken by the providers. Moreover, to ensure a seamless relocation, we consider both the stakeholders' preferences in the process, i.e., users and CSPs. We provide minimum migration overheads in reduced migration time and service downtime from a user's perspective. We also aim to minimize the per-*bit* migration energy consumed during migration from a CSP's standpoint. Therefore, we propose an architecture called *LEAWM* to optimize the migration delay and power jointly. We first define the constraints for finding an effective remapping solution and formulate an optimization problem that yields the three objectives mentioned earlier.

A. Constraints

The constraints corresponding to a feasible service migration are as folows.

1) Completeness of Allocation: Given a VM V_m of a multi-tier application, we define $x(V_m, h_j^k)$ be a binary indicator variable as follows.

$$x(V_m, h_j^k) = \begin{cases} 1 : \text{If } V_m \text{ can is assigned a host } h_j^k \\ 0 : \text{otherwise} \end{cases}. \tag{9}$$

As already defined in Section III-A1, a multi-tier application comprises multiple dependent VMs executing different tasks [13]. Therefore, let $\mathbb{V}_n$ capture the VM request set of a multi-tier application request n. The completeness of allocation can be ensured as per (10).

$$|\mathbb{V}_n| = \sum_{\forall V_m \in \mathbb{V}_n} \min\left(1, \sum_{\forall dc_j \in \mathbb{N}^{DC}} \sum_{\forall h_j^k \in \mathbb{H}_j} x(V_m, h_j^k)\right). \tag{10}$$

Here $|.|$ denotes the cardinality of the set. The above constraints indicate that a feasible allocation exists for all the VMs of a multi-tier application n, thereby implying a complete allocation of the corresponding application.

B. Objectives of Relocation

The optimal service migration is expressed as a bi-objective optimization problem, which is discussed next.

1) Objective 1: Minimizing Migration Delay: The migration delay is to be minimized to improve the quality-of-service (QoS) and responsiveness of applications. Migration delay refers to the total time taken by a VM to seamlessly transfer its state, i.e., memory and CPU states, from the source server to a destination server over a migration path. Let us assume $\mathbb{T}_m^{mig}$ be the migration delay for migrating a VM $V_m \in \mathbb{V}'_n$ on a migration path P_m. The total migration delay for migrating multiple such VMs across DCs can be computed as $\sum_{n\in[1,N]} \sum_{\forall V_m \in \mathbb{V}'_n} \mathbb{T}_m^{mig}$. The overall objective of *LEAWM* is to reduce the overall migration delay incurred by all the applications during the relocation process.

2) Objective 2: Minimizing Migration Energy: From a CSPs perspective, the migration of VMs is an energy-draining process, especially at the CRs encompassing different ISPs. Moreover, these ISPs procure electricity from different providers that experience temporal variations in their prices. This adds to the complexity of deciding the migration paths that encompass multiple such CRs between the source and destination DCs of a VM. The aggregated energy cost can be calculated as $\sum_{n\in[1,N]} \sum_{\forall V_m \in \mathbb{V}'_n} \mathbb{T}_m^{pow}$.

C. Bi-Objective Formulation

The remapping function $\Gamma_R(.)$ is desirable if it can optimize the migration delay and energy consumed in relocating multi-tier application requests. Summarizing the constraints and objectives discussed above, the overall objective of *LEAWM* can be represented as

$$\min \sum_{n\in[1,N]} \sum_{\forall V_m \in \mathbb{V}'_n} (\mathbb{T}_n^{mig} + \mathbb{T}_n^{pow}), \tag{11a}$$

such that

$$V_m^{CPU} \leq h_j^{k.CPU},$$

$$V_m^{Mem} \leq h_j^{k.Mem} \tag{11b}$$

$$\alpha_{m,j} = 1 \tag{11c}$$

$$\sum_{\forall V_m \in \mathbb{V}_n} \mathbb{T}_m^{mig} \leq \delta_n \tag{11d}$$

$$\forall m \in [1, M];\ \forall k \in [1, K]; \forall n \in [1, N]; \forall j \in [1, |\mathbb{N}^{DC}|]. \tag{11e}$$

V. Modeling Migration Delay

Predominantly, the migration involved in *LEAWM* consists of migrating multiple VMs over the WAN links. Various techniques are available in VM migration literature, but we consider pre-copy-based VM migration due to its robustness. The pre-copy-based migration process generally involves image copy followed by an iterative memory copying phase [32], [33]. First, the operating system image is copied from the source server to a target server in a separate IDC in a single iteration. This is followed by an iterative memory copy phase consisting of multiple rounds. In the first round, the entire memory of a VM is transmitted, whereas, in subsequent rounds, memory dirtied in the previous round is transmitted. The process is repeated until a threshold on the remaining dirtied memory or the maximum number of iterations is met [34]. This indicates the beginning of the stop-and-copy phase, where the source VM is suspended, and the leftover dirty pages are copied to the destination in one final round. This marks the end of the handover process, and the VM at the destination takes over the execution, and the source VM is destroyed. As *LEAWM* involves migrating multiple VMs of a given application, scheduling the migration is an inherent challenge. A naive scheduling approach can deliriously impact migration overheads. Before delving into the details of the solution approach, we shed light on the migration strategy used to make the description lucid.

Let $n_{\max}$ be the maximum number of pre-copy iterations. Let V_m^{mem} and V_m^{img} be the memory and image size (in *GB*) of V_m to be transferred. Let $\mathcal{V}_{th}$ be a fixed dirty memory threshold (in *GB*) for stopping the memory copying phase and initiating the stop-and-copy phase. The provisioned bandwidth (in *Gbps*) available for migrating V_m over the migration path P_m is calculated as per (7) and is denoted by $\mathcal{R}_m$. The memory page size (in *KB*) and memory page dirtying rate (pages/second) are considered to be constants for V_m and are denoted as $\wp_m$ and Υ_m respectively. The dirtying rate refers to the number of pages dirtied or updated per second. Then, we define λ_m as the ratio of dirtying rate to the transmission rate, and it can be computed as $\lambda_m = (\wp_m * \Upsilon_m)/\mathcal{R}_m\ ;\quad 0 < \lambda_m < 1$. The VM image is generally copied in a single round, and its duration can be calculated as per (12).

$$\mathbb{T}_m^{img} = \frac{V_m^{img}}{\mathcal{R}_m}. \tag{12}$$

The memory copy phase follows the image copy phase. The memory transmitted for V_m in round i denoted by $\mathcal{M}_m^i$ of this phase can be derived as per (13).

$$\mathcal{M}_m^i = \begin{cases} V_m^{mem} & \text{if } i = 0, \\ \lambda_m * \mathcal{M}_m^{i-1} & \text{otherwise} \end{cases}. \tag{13}$$

From (13), the total migration time $\mathbb{T}_m^{pc+sc}$ (where, $\mathbb{T}_m^{pc+sc} = \mathbb{T}_m^{pc} + \mathbb{T}_m^{sc}$) for the iterative-memory copying of a VM can be calculated as (14). Here, i refers to the current pre-copy round.

$$\mathbb{T}_m^{pc+sc} = \frac{V_m^{mem}}{\mathcal{R}_m}\left(\frac{1-(\lambda_m)^{i_{\max}+1}}{1-\lambda_m}\right). \tag{14}$$

The service degradation time, i.e., downtime or the time taken for the stop-and-copy phase can be calculated as per (15).

$$\mathbb{T}_m^{sc} = \frac{V_m^{mem}}{\mathcal{R}_m}(\lambda_m)^{i_{\max}}. \tag{15}$$

The maximum number of pre-copy rounds $i_{\max}$ can be derived as per (16), where $n_{\max}$ denotes the maximum pre-copy iterations.

$$i_{\max} = \min\left(\left\lceil \log_{\lambda_m} \frac{\mathcal{V}_{th}}{V_m^{mem}} \right\rceil, n_{\max}\right). \tag{16}$$

The total migration time $\mathbb{T}_m^{mig}$ for migrating a VM can be calculated as the sum of the duration of image copy and iterative memory copying rounds. Hence, from (12) and (14), we get,

$$\mathbb{T}_m^{mig} = \mathbb{T}_m^{img} + \mathbb{T}_m^{pc+sc}. \tag{17}$$

Most multi-tier applications have multiple VMs at their disposal. As previously discussed, the CSPs often relocate multiple VMs of different multi-tier applications. Therefore it is of utmost importance to schedule such migrations as a naive migration strategy that can cause severe performance penalties in terms of high migration time and downtime, which can lead to service degradation and user dissatisfaction [35]. Based on previous discussions on pre-copy earlier in this section, we shed some light on the migration strategy incorporated in *LEAWM*.

A. *Migration Strategy for* LEAWM

The authors in [17], discussed a sequential migration strategy to migrate 'M' VMs from a source machine to a destination machine. The sequential migration strategy schedules the migration of VMs serially. The procedure operates as follows. The first migrating VM executes its pre-copy rounds. As the stop-and-copy phase of this VM is triggered, the remaining $M-1$ VMs are suspended and copied in a single iteration. This event of suspension prevents the VMs from dirtying their pages and the pages of other VMs. We adopt this serial migration strategy for migrating multiple VMs and the migration time $\mathbb{T}_s^{pc+sc}$ and downtime $\mathbb{T}_s^{sc}$ denoted for this scheme is computed as per (18) and (19).

$$\mathbb{T}_s^{pc+sc} = \sum_{m=1}^{M} \mathbb{T}_m^{pc+sc} = \frac{MV_m^{mem}}{\mathcal{R}_m}\left(\frac{1-(\lambda_m)^{i_{\max}+1}}{1-\lambda_m}\right) \tag{18}$$

$$\mathbb{T}_s^{sc} = \frac{V_m^{mem}}{\mathcal{R}_m}(\lambda_m)^{i_{\max}} + (M-1)\frac{V_m^{mem}}{\mathcal{R}_m}. \tag{19}$$

VI. Modeling Migration Power

The most critical issue affecting migration performance is the path selection for migrating VMs across geographically distributed IDCs. Since multiple such paths exist for migrating VMs across IDCs, it exhibits temporal variation in electricity price, leading to different electricity costs. Ideally, the CSPs would like to build a migration plan that can minimize migration time and electricity costs. Next, we develop a power consumption model for migrating a single VM across Geo-distributed DCs and extend it to multiple VMs. Let, $\mathcal{Q}_m$ be the set of ISPs to be traversed for migrating V_m along the selected path P_m and $|\mathcal{Q}_m|$ denotes the number of ISP networks visited from $\Gamma_M(V_m)$ to $\Gamma_R(V_m)$ along the path. The energy per *bit* transport in an ISP network $q \in \mathcal{Q}_m$ is represented as $\mathcal{P}_q^{bit}$. Let c_q^t be the cost of electricity at ISP q at *t*th hour of the day. The number of bits transmitted during the migration of V_m for a given iteration i is provided by $\mathcal{M}_m^i$, and it varies depending on the migration strategy. The total network power consumption cost $\mathcal{C}_m^{net}$ for migrating V_m can be computed as per (20).

$$\mathcal{C}_m^{net} = \sum_{i=1}^{i_{\max}} \sum_{q=1}^{|\mathcal{Q}_m|} \mathcal{M}_m^i * \mathcal{P}_q^{bit} * c_q^t. \tag{20}$$

Let $\mathcal{C}_D$ denote a constant power cost incurred at the source and destination hosts. By ignoring the additional administrative overhead for connection setup, location selection, etc., the aggregate power consumption cost for migrating V_m can be calculated as per (21).

$$\mathcal{C}_m^{int} = \mathcal{C}_D + \mathcal{C}_m^{net}. \tag{21}$$

Hence, for a given migration path, the total power consumption cost $\mathbb{T}_m^{pow}$ for migrating V_m over interconnection DCs can be expressed as per (22), where β is an additional overhead involved in migration for path establishment and processing of data at intermediate nodes.

$$\mathbb{T}_m^{pow} = \mathcal{C}_m^{int}(1+\beta). \tag{22}$$

VII. Solution Using ACO Meta-Heuristic

Given the initial allocation Γ_M, and an alternative remapping Γ_R, the primary objective of *LEAWM* is to build a feasible relocation plan that finds suitable paths to relocate multiple VMs of multi-tier applications respecting the location constraints and resource demands such that the overall migration delay and migration energy is reduced. However, finding such a solution to the relocation problem is $\mathcal{NP}$-Hard [36]. Hence we use an Ant colony optimization (ACO) based solution approach to obtain a Pareto-efficient solution within acceptable time limits. ACO is a robust meta-heuristic-based algorithm suitable for solving complex practical problems that are computationally expensive [37]. Specific to cloud computing, Gao et al. [38] proposed a multi-objective ant colony system algorithm-based VM placement to simultaneously minimize the total resource wastage and power consumption of servers hosting VMs. In [39] is presented an ACO-based distributed system to perform dynamic VM consolidation to reduce energy consumption at cloud DCs while adhering to the application QoS. On the other hand, a novel algorithm based on ACO is developed in [40] to solve the virtual network embedding (VNE) problem specifically, the authors focused on virtual link mapping onto substrate networks. The above implementations showed promising performance and achieved proper trade-offs among multiple parameters. Motivated by this, we propose to utilize ACO to strike a trade-off between migration time and migration power.

Following the principles of ACO, we consider a relocation plan of V_m as a path P_m from its source host $\Gamma_M(V_m)$ to $\Gamma_R(V_m)$ encompassing multiple core routers of different ISPs that obtain services from disparate vendors having temporal variation in electricity price. This problem is analogous to the traveling salesman problem (TSP), relaxing the constraint that demands the artificial ants to visit all the graph nodes. The path traversed is the migration path for the corresponding VM, and the start and the end vertex denote the initial and remapped servers, respectively. An artificial agent, i.e., an ant, traverses the substrate graph to find a feasible relocation. Next, we define the two most important parameters involved in the working of ACO, i.e., the pheromone trial matrix and heuristic information matrix. Let this heuristic information be represented as η, and pheromone trial information is ζ. The heuristic information between nodes f and g along the path denoted by $\eta_{f,g}$ indicates the desirability of an ant currently at node f in selecting g as its next node for traversal. Similarly, the pheromone trail $\zeta_{f,g}$ guides the ants in selecting the next node that complies with the global objective function. Next, we discuss the step-wise working of ACO.

A. Initial Pheromone Trial Determination

The initial pheromone trail ζ_0 is calculated as per (23). Here, $\mathcal{S}_0$ be the solution obtained by random path selection technique; $\mathbb{T}^{mig}(\mathcal{S}_0)$ and $\mathbb{T}^{pow}(\mathcal{S}_0)$ denote the migration time and energy cost, respectively, of $\mathcal{S}_0$.

$$\zeta_0 = \frac{1}{(\mathbb{T}^{mig}(\mathcal{S}_0) + \mathbb{T}^{pow}(\mathcal{S}_0))}. \tag{23}$$

B. Computing Heuristic Information

An essential aspect of ACO-based solution strategies is choosing a good heuristic, which in combination with the pheromone information, helps generate efficient solutions. The heuristic information acts as a guide for the ACO with problem-specific knowledge. As the overall objective is to construct a solution to minimize both migration time and migration power, the heuristic functions are bi-objective and are expressed as follows.

$$\eta_{f,g,1} = \frac{1}{\sum_{m=1}^{|V_n'|} \mathbb{T}_m^{mig}}; \qquad \eta_{f,g,2} = \frac{1}{\sum_{m=1}^{|V_n'|} \mathbb{T}_m^{pow}}. \tag{24}$$

The heuristic information denoted by $\eta_{f,g}$ indicates the desirability of an ant currently located at node f to move to node g and is expressed as per the following equation.

$$\eta_{f,g} = \eta_{f,g,1} + \eta_{f,g,2}. \tag{25}$$

Equation (25) indicates the desirability of an ant currently at node f to dynamically select node g in the traversal path that minimizes the migration delay and power in the remapping Γ_R. The nodes f and g reflect the intermediate devices along ISP paths. In fact, the heuristic information as in (25) captures partial contribution of a movement from the current node f to the next node g towards the objective function as realized in (11a) and (11b).

C. Computing Pheromone Trials

The pheromones deposited can increase as more ants traverse an edge or decrease due to vaporization. New deposits of pheromones make sure that better solutions are retained. ACO comprises two different pheromone updating rules (i.) local update and (ii.) global update rules.

1) Local Update Rule: The local update rule is defined as per (26), where Λ_{lc} denotes the local pheromone decay parameter and lies in the range of $0 < \Lambda_{lc} < 1$.

$$\zeta_{f,g} = (1 - \Lambda_{lc})\zeta_{f,g} + \Lambda_{lc}\Delta\zeta_{f,g}. \tag{26}$$

2) Global Update Rule: The global update is performed once all ants have constructed their respective solutions. It is done to make the edges traversed in the best solution's path more desirable in the subsequent rounds. The global update rule is defined as per (27). λ_{gb} denotes the global pheromone evaporation parameter and lies in the range of $0 < \Lambda_{gb} < 1$. $\mathbb{S}_{opt}$ represents the best solution generated in the current iteration. $\mathbb{T}^{mig}(\mathcal{S}_{opt})$ and $\mathbb{T}^{pow}(\mathcal{S}_{opt})$ denote the migration time and energy cost of the best solution.

$$\zeta_{f,g} = (1 - \Lambda_{gb})\zeta_{f,g} + \Lambda_{gb}\Delta\zeta_{f,g}, \tag{27}$$

$\Delta\zeta_{f,g}$ can be calculated as per the following equation.

$$\Delta\zeta_{f,g} = \begin{cases} (\mathbb{T}^{mig}(\mathcal{S}_{opt})^{-1}.\mathcal{T}^{pow}(\mathcal{S}_{opt})^{-1}) & \forall e_l \in \mathcal{S}_{opt} \\ 0 & \text{otherwise} \end{cases}. \tag{28}$$

D. Iterative Solution Construction

For an ant z, constructing a migration path currently at node f selects g as its next node for visiting based on the state transition rule defined in (29). α is a parameter that denotes the relative importance of the pheromone trail and heuristic information. r denotes a random number uniformly distributed in the range of $[0, 1]$. The parameter r_0 is fixed and lies between $0 \leq r_0 \leq 1$ and S is random number selected as per (30). $\omega(z)$ is a set of nodes that have not been visited by ant z yet.

$$g = \begin{cases} \arg\max_{g\in\omega(z)} \{\alpha[\zeta_{f,g}].(1-\alpha)[\eta_{f,g}]\} \\ \qquad \text{if } r < r_0 \text{ (Exploit)} \\ S \quad \text{otherwise (Biased exploration)} \end{cases} \tag{29}$$

$$p_{f,g}^{z} = \begin{cases} \frac{\alpha[\zeta_{f,g}].(1-\alpha)[\eta_{f,g}]}{\sum\limits_{d\in\omega(z)} \alpha[\zeta_{f,d}].(1-\alpha)[\eta_{f,d}]} & \text{if } j \in \omega(z) \\ 0 & \text{otherwise} \end{cases}. \tag{30}$$

The overall state transition rule from (29) and (30) is called pseudo-random proportionality rule, which favors selecting the next node with lower power consumption and migration time. The parameter r_0 dictates the importance of exploration versus exploitation. In the first case, where $r < r_0$, ants exploit the best solution following (29); otherwise the next node for traversal is randomly selected based on (30). We stop the iterations when two consecutive iterations yield little improvement.

VIII. Performance Evaluation

We have used the CloudSim simulation toolkit[1] to implement the proposed mechanism and analyzed its performance via simulation considering different test cases comprising [10, 25, 50, 100] multi-tier applications. For simulation, we have considered 4 geographically separated DCs. Each DC is implemented as a three-layered fat-tree topology [41], [42]. The

[1][Online]. Available: http://www.cloudbus.org/cloudsim/ (Accessed: 2023/05/18 19:08:24)

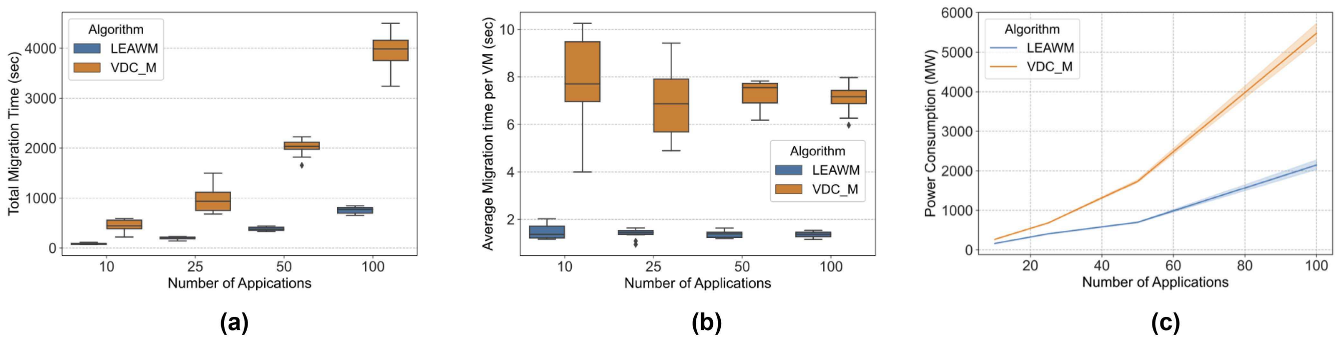

Fig. 3. (a) Total Migration time (*sec*) versus Number of Applications, (b) Average Migration Time (*sec*) versus Number of Applications, (c) Power Consumption (*MW*) versus Number of Applications.

TABLE II
KEY PARAMETERS [13], [15]

Parameter	Value
Number of Multi-Tier Applications	$[10, 25, 50, 100]$
Number of Tiers	3
VMs per Tier	$U[1, 3]$
Dirtying Rate of a VM	$U(0, 1)$
Memory of a VM	$U(1, 8)$
Number of VCPUs of a VM	$U(1, 4)$
Number of DCs	4
Number of servers per DC	16
Server's VCPUs	$U(1, 16)$
Server's Memory	$U(1, 32)$
Capacity of ToR links	1 *Gbps*
Capacity of Switch to Switch links	40 *Gbps*
Capacity of Inter DC Links	40 *Gbps*
$\mathcal{P}_{bit}$	$1.108 * 10^{-5} W/bit$
β	0.5
$\mathcal{C}_D$	12.5 *W*

geographical separation leads to different IDCs getting served by different electricity providers, and their prices vary hourly. However, we assume that the electricity price does not change during migration and is triggered after considering the latest change in price. The rest of the simulation parameters are set following Table II.

A. The Baseline Algorithm

To compare the performance of *LEAWM*, we consider a modified version of the heuristic-based algorithm in [15] (referred to as VDC_M) as a baseline. VDC_M aims to reduce virtual data center (VDC) re-embedding costs considering the cost of hosting VMs on servers and virtual links over substrate paths. We consider migrating multi-tier applications instead of VDC requests using a parallel migration algorithm to compare the performance and ignore the virtual link mapping cost as a parameter for evaluation.

B. Experimental Results

The performance of *LEAWM* is evaluated from three different perspectives. First, its performance is gauged depending on the trade-off achieved between migration delay and migration power. Second, we assess the performance of *LEAWM* based on migration-related metrics, e.g., memory copied, the number of iterations, and application downtime. Finally, we evaluate overheads of *LEAWM* in terms of metrics like the convergence rate and execution time.

1) Migration Delay and Migration Power Trade-Off: Fig. 3(a) shows the variation of migration time for $[10, 25, 50, 100]$ applications for *LEAWM* and VDC_M. It can be observed that the aggregate migration time increases for an increasing number of applications, which is expected behavior. We also observe that *LEAWM* can significantly reduce migration time compared to the baseline. This is attributed to the fact that *LEAWM* strategy based on a meta-heuristic ACO can find remapping paths that lead to reduced migration delay. The migration strategy for *LEAWM* can find out shorter migration paths with higher available bandwidth. Similar behavior is observed considering the average migration time of individual VMs and is depicted in Fig. 3(b).

Fig. 3(c) shows the variation in the power consumption for the strategies used for comparison for an increasing number of applications. *LEAWM* consumes less power compared to the baseline algorithm. The improved performance is achieved due to the following reasons: (*i*) *LEAWM* is able to find shorter paths, (*ii*) it is also able to find a shorter path with maximum available bandwidth which directly reduces the migration delay, (*iii*) owing to the shorter migration paths less memory is transmitted during the migration process. This can be directly inferred from Fig. 5(a). Since the power consumption is directly related to the amount of memory copied (refer to (20), the proposed technique performs superior compared to the baseline. Moreover, the higher the transmission data, the higher energy consumed at the intermediate devices, thereby having a deleterious impact on power consumption.

2) Migration-Related Metrics: This section discusses and compares the performance of *LEAWM* and VDC_M using migration-specific metrics. First, we compare the performance using application downtime, which refers to the time taken by the stop-and-copy phase when the services are halted. Fig. 4(a) and (b) denote the total downtime and average downtime for different number of applications across the two different techniques. It can be observed from Fig. 4(a) that the total application downtime increases with an increasing number of applications. This is because the amount of memory copied in the stop-and-copy phase of *LEAWM* is lower than VDC_M. This is due to the

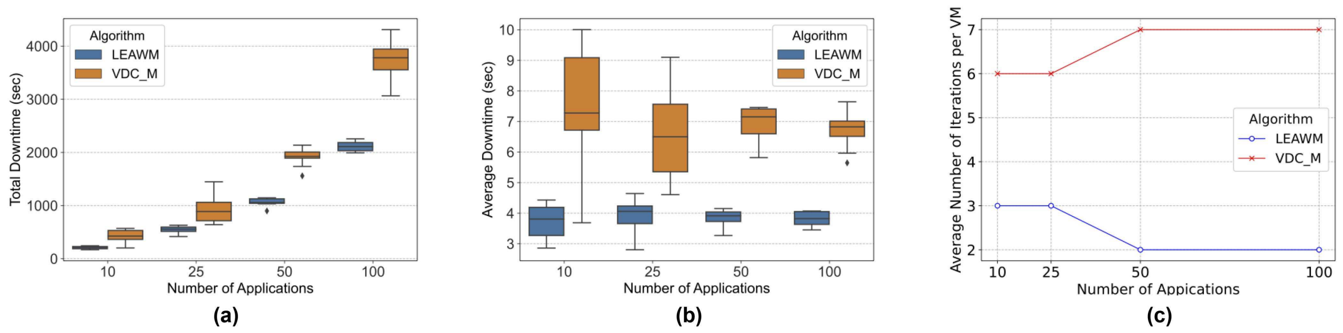

Fig. 4. (a) Total Application Downtime (*sec*) versus Number of Applications, (b) Average Application Downtime (*sec*) versus Number of Applications, (c) Average number of Iterations versus Number of Applications.

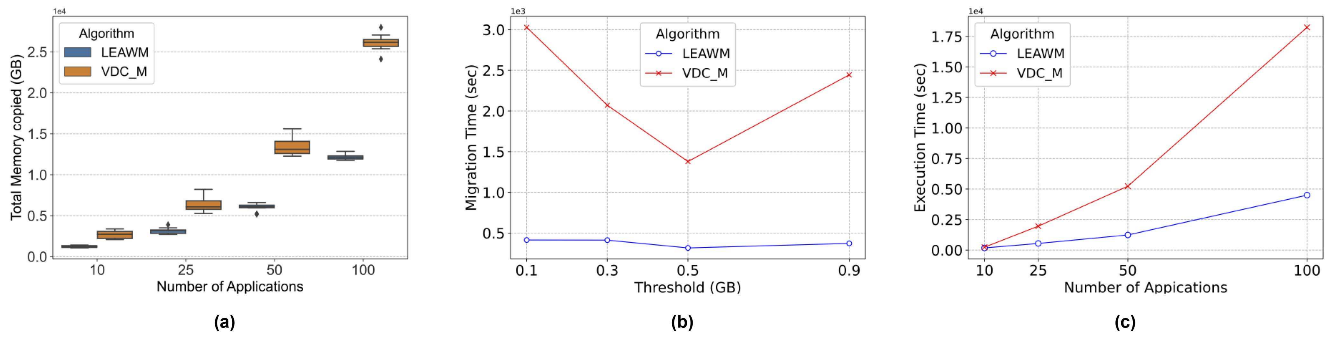

Fig. 5. (a) Total Memory Copied (*GB*) versus Number of Applications, (b) Total Migration Time for 50 Applications versus Pre-copy Migration Memory Threshold, (c) Execution Time (*sec*) versus Number of Applications.

inability of the migration protocol of VDC_M to map to find migration paths with a lower number of hops with higher bandwidth leading to elevated downtime. Additionally, forceful termination of VMs at $n_{\max}$ iterations due to the shared bandwidth among multiple migrating VMs leads to more memory being copied in the stop-and-copy phase, thereby contributing to increased downtime. Similar conclusions can be drawn from Fig. 4(b) that denotes the average downtime suffered by each VM, and it is significantly lower in comparison with the baseline. Considering the amount of memory transmitted in the migration process, the strategies' behavior is captured in Fig. 5(a).

It can be observed that *LEAWM* transmits less memory compared to the baseline algorithm. This behavior is attributed to the fact that *LEAWM* is not only able to find a shorter path with higher migration bandwidth but also is able to allocate the entire migration bandwidth to individual VMs in contrast to VDC_M that fails to perform both. VDC_M provides a limited share of the migration bandwidth for VMs, thereby prolonging the convergence of pre-copy migration latency. This can be realized from Fig. 4(c). It can be seen that the number of iterations is higher in the case of VDC_M, thereby leading to higher pre-copy rounds, which leads to higher migration time and power consumed. Moreover, it can also be observed that for a more significant test case, i.e., 50 and 100, the pre-copy rounds are forcefully terminated at $n_max = 8$. This is because the pre-copy never converges as the suitable migration bandwidth required is never achieved and pre-copy is prematurely halted, leading to more memory being transferred in the stop-and-copy and elevating the downtime.

Fig. 5(b) reflects the variation of the migration delay for variation in the threshold memory for both strategies used for comparison. We can observe that the total delay is higher for VDC_M in comparison with *LEAWM*. However, we observe an interesting behavior considering VDC_M. The migration delay decreases till 0.5 GB as the pre-copy converges earlier attributed to the higher threshold, thereby reducing iterations and reducing migration time. Beyond 0.5, the migration delay escalates as pre-copy converges very quickly, leading to almost the entire memory being transmitted in approximately one round, causing a bandwidth bottleneck for VMs, elevating the migration delay.

3) Other Performance Metrics: This section discusses some other performance metrics, such as execution time and convergence rate. Fig. 5(c) shows the execution time comparison for *LEAWM* and the baseline algorithm VDC_M. The baseline algorithm consumes more time as it is brute force, and it tests all feasible solutions before deciding on the best solution. Since the search space for such a problem increases exponentially with increasing problem size (Number of applications), the baseline algorithm consumes more time.

Fig. 6 shows the convergence rate for migrating 50 applications for *LEAWM*. The normalized cost is obtained following [43]. We observe from the plot that the proposed ACO-based algorithm's convergence rate is reasonable, and it quickly converges to an optimal solution (close to 400 iterations for

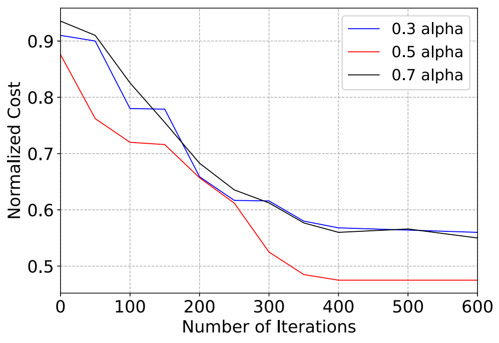

Fig. 6. Convergence rate.

all cases). Similar observations were made for 10, 25, and 100 applications; hence we omit such plots.

From the results, we observe that *LEAWM* can reduce both migration delay and power consumed during the remapping of multi-tier applications across Geo-distributed IDCs, interconnected over a backbone network. Further, it can also be inferred from the results that the proposed scheme can reduce the migration delay by almost 25%–30% of what is observed for the greedy approach. In comparison, we follow a close to 25% reduction in the power cost compared to the baselines.

IX. Conclusion

In this article, we have presented a novel approach, namely *LEAWM* for migration path selection for VM migrations over geo-distributed clouds that spans over multiple DCs. We optimize the migration delay and migration power due to the migration traffic and develop a bi-objective optimization technique based on an ant colony meta-heuristic. We have implemented and tested the proposed system over a simulation framework. The results show that the proposed approach can yield a 25–30% reduction in the VM migration delay while ensuring close to 20% reduction in the power consumption compared to the baseline algorithm. As a future extension of this work, we plan to extend the proposed technique for a federated architecture that includes multiple ISPs connected via a more complex backbone network.

Acknowledgments

The authors would like to thank the reviewers for insightful comments to help improve the manuscript quality.

References

[1] W. Wu, W. Wang, X. Fang, J. Luo, and A. V. Vasilakos, "Electricity price-aware consolidation algorithms for time-sensitive VM services in cloud systems," *IEEE Trans. Serv. Comput.*, vol. 14, no. 6, pp. 1726–1738, Nov./Dec. 2021.

[2] S. Rahman, A. Gupta, M. Tornatore, and B. Mukherjee, "Dynamic workload migration over backbone network to minimize data center electricity cost," *IEEE Trans. Green Commun. Netw.*, vol. 2, no. 2, pp. 570–579, Jun. 2018.

[3] C. Gu, Z. Li, H. Huang, and X. Jia, "Energy efficient scheduling of servers with multi-sleep modes for cloud data center," *IEEE Trans. Cloud Comput.*, vol. 8, no. 3, pp. 833–846, Third Quarter 2020.

[4] T. N. Le, J. Liang, Z. Liu, R. K. Sitaraman, J. Nair, and B. J. Choi, "Optimal energy procurement for geo-distributed data centers in multi-timescale electricity markets," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 45, no. 3, pp. 185–197, 2018.

[5] S. Ahmad, A. Rosenthal, M. H. Hajiesmaili, and R. K. Sitaraman, "Learning from optimal: Energy procurement strategies for data centers," in *Proc. 10th ACM Int. Conf. Future Energy Syst.*, 2019, pp. 326–330.

[6] S. Albers and J. Quedenfeld, "Optimal algorithms for right-sizing data centers," in *Proc. 30th Symp. Parallelism Algorithms Architectures*, 2018, pp. 363–372.

[7] A. Marotta, S. Avallone, and A. Kassler, "A joint power efficient server and network consolidation approach for virtualized data centers," *Comput. Netw.*, vol. 130, pp. 65–80, 2018.

[8] B. Heller et al., "ElasticTree: Saving energy in data center networks," in *Proc. 7th USENIX Conf. Netw. Syst. Des. Implementation*, 2010, pp. 249–264.

[9] A. O. F. Atya, Z. Qian, S. V. Krishnamurthy, T. La Porta, P. McDaniel, and L. M. Marvel, "Catch me if you can: A closer look at malicious co-residency on the cloud," *IEEE/ACM Trans. Netw.*, vol. 27, no. 2, pp. 560–576, Apr. 2019.

[10] H. Lu, C. Xu, C. Cheng, R. Kompella, and D. Xu, "vHaul: Towards optimal scheduling of live multi-VM migration for multi-tier applications," in *Proc. IEEE 8th Int. Conf. Cloud Comput.*, 2015, pp. 453–460.

[11] S. K. Addya, A. Satpathy, B. C. Ghosh, S. Chakraborty, and S. K. Ghosh, "Power and time aware VM migration for multi-tier applications over geo-distributed clouds," in *Proc. IEEE 12th Int. Conf. Cloud Comput.*, 2019, pp. 339–343.

[12] T. He, A. N. Toosi, and R. Buyya, "CAMIG: Concurrency-aware live migration management of multiple virtual machines in SDN-Enabled clouds," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 10, pp. 2318–2331, Oct. 2022.

[13] S. K. Addya, A. Satpathy, B. C. Ghosh, S. Chakraborty, S. K. Ghosh, and S. K. Das, "CoMCLOUD: Virtual machine coalition for multi-tier applications over multi-cloud environments," *IEEE Trans. Cloud Comput.*, vol. 11, no. 1, pp. 956–970, First Quarter 2023.

[14] N. Tziritas, T. Loukopoulos, S. U. Khan, C.-Z. Xu, and A. Y. Zomaya, "Online live VM migration algorithms to minimize total migration time and downtime," in *Proc. IEEE Int. Parallel Distrib. Process. Symp.*, 2019, pp. 406–417.

[15] G. Sun, D. Liao, D. Zhao, Z. Xu, and H. Yu, "Live migration for multiple correlated virtual machines in cloud-based data centers," *IEEE Trans. Serv. Comput.*, vol. 11, no. 2, pp. 279–291, Mar./Apr. 2018.

[16] U. Mandal, P. Chowdhury, M. Tornatore, C. U. Martel, and B. Mukherjee, "Bandwidth provisioning for virtual machine migration in cloud: Strategy and application," *IEEE Trans. Cloud Comput.*, vol. 6, no. 4, pp. 967–976, Fourth Quarter 2018.

[17] G. Sun, D. Liao, V. Anand, D. Zhao, and H. Yu, "A new technique for efficient live migration of multiple virtual machines," *Future Gener. Comput. Syst.*, vol. 55, pp. 74–86, 2016.

[18] H. Jin, L. Deng, S. Wu, X. Shi, H. Chen, and X. Pan, "MECOM: Live migration of virtual machines by adaptively compressing memory pages," *Future Gener. Comput. Syst.*, vol. 38, pp. 23–35, 2014.

[19] F. Callegati and W. Cerroni, "Live migration of virtualized edge networks: Analytical modeling and performance evaluation," in *Proc. IEEE SDN Future Netw. Serv.*, 2013, pp. 1–6.

[20] S. Sahni and V. Varma, "A hybrid approach to live migration of virtual machines," in *Proc. IEEE Int. Conf. Cloud Comput. Emerg. Markets*, 2012, pp. 1–5.

[21] M. R. Hines, U. Deshpande, and K. Gopalan, "Post-copy live migration of virtual machines," *ACM SIGOPS Operating Syst. Rev.*, vol. 43, no. 3, pp. 14–26, 2009.

[22] M. F. Bari, M. F. Zhani, Q. Zhang, R. Ahmed, and R. Boutaba, "CQNCR: Optimal VM migration planning in cloud data centers," in *Proc. IFIP Netw. Conf.*, 2014, pp. 1–9.

[23] A. Qureshi, R. Weber, H. Balakrishnan, J. Guttag, and B. Maggs, "Cutting the electric bill for internet-scale systems," in *Proc. ACM SIGCOMM Conf. Data Commun.*, 2009, pp. 123–134.

[24] A. Gupta, U. Mandal, P. Chowdhury, M. Tornatore, and B. Mukherjee, "Cost-efficient live VM migration based on varying electricity cost in optical cloud networks," *Photonic Netw. Commun.*, vol. 30, no. 3, pp. 376–386, Dec. 2015.

[25] L. Rao, X. Liu, L. Xie, and W. Liu, "Minimizing electricity cost: Optimization of distributed internet data centers in a multi-electricity-market environment," in *Proc. IEEE Conf. Comput. Commun.*, 2010, pp. 1–9.

[26] M. A. Adnan, R. Sugihara, and R. K. Gupta, “Energy efficient geographical load balancing via dynamic deferral of workload,” in *Proc. IEEE 5th Int. Conf. Cloud Comput.*, 2012, pp. 188–195.
[27] F. Zhang, G. Liu, X. Fu, and R. Yahyapour, “A survey on virtual machine migration: Challenges, techniques, and open issues,” *IEEE Commun. Surveys Tuts.*, vol. 20, no. 2, pp. 1206–1243, Second Quarter 2018.
[28] H. Jin, D. Pan, J. Xu, and N. Pissinou, “Efficient VM placement with multiple deterministic and stochastic resources in data centers,” in *Proc. IEEE Glob. Commun. Conf.*, 2012, pp. 2505–2510.
[29] M. Mishra and U. Bellur, “Whither tightness of packing? The case for stable VM placement,” *IEEE Trans. Cloud Comput.*, vol. 4, no. 4, pp. 481–494, Fourth Quarter 2016.
[30] Regions and availability zones, Feb. 2022. [Online]. Available: https://aws.amazon.com/about-aws/global-infrastructure/regions_az/?p=ngi&loc=2
[31] X. Yi, F. Liu, J. Liu, and H. Jin, “Building a network highway for Big Data: Architecture and challenges,” *IEEE Netw.*, vol. 28, no. 4, pp. 5–13, Jul./Aug. 2014.
[32] V. Kherbache, E. Madelaine, and F. Hermenier, “Scheduling live migration of virtual machines,” *IEEE Trans. Cloud Comput.*, vol. 8, no. 1, pp. 282–296, First Quarter 2020.
[33] R. Torre, R.-S. Schmoll, F. Kemser, H. Salah, I. Tsokalo, and F. H. Fitzek, “Demo: Benchmarking live migration performance of two trendy virtualization technologies,” in *Proc. IEEE Conf. Comput. Commun. Workshops*, 2020, pp. 1087–1088.
[34] S. K. Addya, A. K. Turuk, A. Satpathy, B. Sahoo, and M. Sarkar, “A strategy for live migration of virtual machines in a cloud federation,” *IEEE Syst. J.*, vol. 13, no. 3, pp. 2877–2887, Sep. 2019.
[35] A. Satpathy, M. N. Sahoo, A. Mishra, B. Majhi, J. J. Rodrigues, and S. Bakshi, “A service sustainable live migration strategy for multiple virtual machines in cloud data centers,” *Big Data Res.*, vol. 25, 2021, Art. no. 100213, doi: 10.1016/j.bdr.2021.100213.
[36] M. Chowdhury, M. R. Rahman, and R. Boutaba, “ViNEYard: Virtual network embedding algorithms with coordinated node and link mapping,” *IEEE/ACM Trans. Netw.*, vol. 20, no. 1, pp. 206–219, Feb. 2012.
[37] Y.-H. Jia et al., “An intelligent cloud workflow scheduling system with time estimation and adaptive ant colony optimization,” *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 51, no. 1, pp. 634–649, Jan. 2021.
[38] Y. Gao, H. Guan, Z. Qi, Y. Hou, and L. Liu, “A multi-objective ant colony system algorithm for virtual machine placement in cloud computing,” *J. Comput. Syst. Sci.*, vol. 79, no. 8, pp. 1230–1242, 2013.
[39] F. Farahnakian et al., “Using ant colony system to consolidate VMs for green cloud computing,” *IEEE Trans. Serv. Comput.*, vol. 8, no. 2, pp. 187–198, Mar./Apr. 2015.
[40] H.-K. Zheng et al., “Link mapping-oriented ant colony system for virtual network embedding,” in *Proc. IEEE Congr. Evol. Comput.*, 2017, pp. 1223–1230.
[41] T. Duong-Ba, T. Tran, T. Nguyen, and B. Bose, “A dynamic virtual machine placement and migration scheme for data centers,” *IEEE Trans. Serv. Comput.*, vol. 14, no. 2, pp. 329–341, Mar./Apr. 2021.
[42] A. Zhou, S. Wang, X. Ma, and S. S. Yau, “Towards service composition aware virtual machine migration approach in the cloud,” *IEEE Trans. Serv. Comput.*, vol. 13, no. 4, pp. 735–744, Jul./Aug. 2020.
[43] F. Haider, D. Zhang, M. St-Hilaire, and C. Makaya, “On the planning and design problem of fog computing networks,” *IEEE Trans. Cloud Comput.*, vol. 9, no. 2, pp. 724–736, Second Quarter 2021.

Sourav Kanti Addya (Senior Member, IEEE) received the PhD degree in CSE from NIT Rourkela, India and worked as a postdoctoral fellow with the Department of CSE, IIT Kharagpur, India. He is currently an assistant professor with the Department of CSE, NITK Surathkal, India. His research interests include cloud system, serverless computing, IoT, blockchain. He is a member of the ACM.

Anurag Satpathy (Graduate Student Member, IEEE) received the BTech degree in information technology from IIIT Bhubaneswar, India, in 2014, the MTech degree in CSE from BIT Mesra, Ranchi, India, in 2017, and the PhD degree in CSE from NIT Rourkela, India. His research interests include cloud computing, the Internet of Things, and distributed systems.

Bishakh Chandra Ghosh (Student Member, IEEE) received the BTech degree in information technology from NIT Durgapur, India. He is a doctoral research student in CSE with IIT Kharagpur. His current research area includes cloud computing, blockchain, and distributed systems.

Sandip Chakraborty (Member, IEEE) received the PhD degree in CSE from IIT Guwahati. Currently, he is an associate professor in CSE with IIT Kharagpur. He is working as an area editor of Elsevier's *Ad Hoc Networks* and *Pervasive and Mobile Computing* journals. His primary research interests are on the intersections of computer systems, distributed systems, and human-computer interaction.

Soumya K. Ghosh (Senior Member, IEEE) received the PhD degree in CSE from the IIT, Kharagpur. Currently, he is a professor with the Department of CSE, IIT Kharagpur. Prior to joining IIT Kharagpur, he was with the ISRO in the area of satellite remote sensing and GIS. His research interests include cloud computing, spatial data science, and IoT.

Sajal K. Das (Fellow, IEEE) is a Curators' distinguished professor of computer science and Daniel St. Clair Endowed chair with the Missouri University of Science and Technology. His research interests include wireless and sensor networks, mobile and pervasive computing, cyber-physical systems, IoT, cloud computing, and cybersecurity. He is the editor-in-chief of Elsevier's *Pervasive and Mobile Computing* journal, and associate editor of *IEEE Transactions on Mobile Computing*, *IEEE Transactions on Dependable and Secure Computing*, and *ACM Transactions on Sensor Networks*. He is a distinguished Alumnus of the Indian Institute of Science, Bangalore.