Computers, Materials & Continua
DOI: 10.32604/cmc.2023.031344
Article

Tech Science Press

A Service Level Agreement Aware Online Algorithm for Virtual Machine Migration

Iftikhar Ahmad[1,*], Ambreen Shahnaz[2], Muhammad Asfand-e-Yar[3], Wajeeha Khalil[1] and Yasmin Bano[1]

[1]Department of Computer Science and Information Technology, University of Engineering and Technology, Peshawar, 25120, Pakistan
[2]Department of Computer Science, Shaheed Benazir Bhutto Women University, Peshawar, 25000, Pakistan
[3]Center of Excellence in Artificial Intelligence, Department of Computer Science, Bahria University, Islamabad, Pakistan
*Corresponding Author: Iftikhar Ahmad. Email: ia@uetpeshawar.edu.pk
Received: 15 April 2022; Accepted: 08 June 2022

Abstract: The demand for cloud computing has increased manifold in the recent past. More specifically, on-demand computing has seen a rapid rise as organizations rely mostly on cloud service providers for their day-to-day computing needs. The cloud service provider fulfills different user requirements using virtualization - where a single physical machine can host multiple Virtual Machines. Each virtual machine potentially represents a different user environment such as operating system, programming environment, and applications. However, these cloud services use a large amount of electrical energy and produce greenhouse gases. To reduce the electricity cost and greenhouse gases, energy efficient algorithms must be designed. One specific area where energy efficient algorithms are required is virtual machine consolidation. With virtual machine consolidation, the objective is to utilize the minimum possible number of hosts to accommodate the required virtual machines, keeping in mind the service level agreement requirements. This research work formulates the virtual machine migration as an online problem and develops optimal offline and online algorithms for the single host virtual machine migration problem under a service level agreement constraint for an over-utilized host. The online algorithm is analyzed using a competitive analysis approach. In addition, an experimental analysis of the proposed algorithm on real-world data is conducted to showcase the improved performance of the proposed algorithm against the benchmark algorithms. Our proposed online algorithm consumed 25% less energy and performed 43% fewer migrations than the benchmark algorithms.

Keywords: Cloud computing; green computing; online algorithms; virtual machine migration

1 Introduction

The demand for cloud computing is increasing tremendously with each successive day. Many leading Information Technology (IT) companies like Amazon, Google, IBM, and Microsoft, have deployed hundreds of data centers to cater to the growing demand of customers for computing resources [1]. The customers benefit by using the IT resources (hardware and software) on a pay-as-you-go model, thus saving high upfront costs [2].

Cloud computing infrastructure houses thousands of computing servers and requires an enormous amount of energy to operate [3]. The energy is required not only for operating servers but is also used for cooling purposes to provide an optimum environment for the computing servers to operate. According to Belady [4], IT costs only contribute 25% of the total cost of a data center, the rest of the 75% costs are incurred by infrastructure and energy. Computing servers in a data center operate at approximately 10%–15% of their maximum capacity and remain idle most of the time [5]. This results in a huge loss in terms of monetary value for the service providers. According to [6] data centers are using 2% of the total US electricity demands, and by using energy efficient techniques 80% of savings can be made. A report published by the U.S. Department of Energy estimated that data centers operating in U.S. consumed approximately 70 billion kilowatt-hours in 2014 [7,8]. The energy consumption of U.S. data centers reached 200 billion kilowatt-hours by the end of 2020 [8].

Besides the economic downsides, the high energy consumption also has a negative impact on the environment. Energy is required not only for operating the servers but also for cooling facilities to provide an optimum environment for servers to operate effectively [6]. Thus, the combination of high electricity demands and cooling requirements cause an increase in carbon emission as well as other toxic gases. The indirect impact includes the usage of brown energy which is generated by burning carbon-intensive fossil fuels [6].

In addition to energy constraints, cloud service providers are also restricted by a Service Level Agreement (SLA) constraint. An SLA is an agreement between a cloud service provider and a customer. The SLA specifies various business conditions between the two parties. Among others, SLA defines the quality of service that the service provider is bound to provide to the customer. SLA also includes Service Level Agreement Violation (SLAV) clause, which stipulates the penalty imposed on the service provider if she fails to provide the agreed upon QoS to the customer.

Therefore, energy efficiency as well as provision of QoS as per agreed SLA, are fundamental considerations in cloud computing environments. The energy utilization of data centers can be lessened by using virtualization technology which is a key element of cloud computing. It provides an abstraction by hiding the complexities of the underlying hardware or software [2,9,10]. The technology increases resource utilization by allowing different virtual machine instances to run on a single host and share the resources without any interference. Fig. 1 depicts a sample physical host with two virtual machines. Virtualization also provides the benefit of virtual machine (VM) migration, which helps in the dynamic consolidation of VMs thus decreasing energy consumption. Dynamic consolidation helps in minimizing the number of active hosts by migrating VMs from underutilized hosts and switching hosts to low power mode, thus reducing the total energy utilization. VMs are also migrated when a host faces performance degradation due to SLAV that occurs when hosts become overloaded and unable to meet the resource requirement of VMs. Therefore, cloud service providers must deal with energy-performance trade-off. Green cloud computing addresses this trade-off by providing efficient utilization of computing resources and minimizing energy consumption which reduces negative environmental impact.

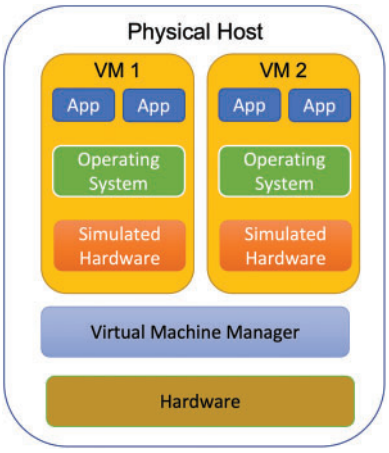

Figure 1: A single host with two virtual machines

Virtual machine migration is a classical online problem. In an online problem, the algorithm does not have access to the future input and processes the input as it arrives. At each time point, the online algorithm is presented with a request and shall take an irreversible decision without information about the future requests [11]. Examples of online problems include currency conversion, cache problem, and portfolio selection problem [11]. In contrast to online algorithms, offline algorithms have complete knowledge about the input instance beforehand and make a decision based on the availability of the complete input instances. Formally, we define an online algorithm as follows [11]. An algorithm **A** computes online if at each time point $t \in [1, 2, \ldots T-1]$, **A** takes a decision at the time t before the input at time $t+1$ is revealed. An algorithm **OFF** computes offline if the complete input instance is available at $t = 1$. **OFF** is called optimum if it produces the best possible solution on any given input instance. In our problem settings, the online algorithm does not have any knowledge about the incoming virtual machines and their processing requirements. Similarly, the algorithm is oblivious if SLAV will occur, and if it occurs what will be the start and end times of SLAV. The unavailability of future information makes the design of online algorithms a challenging task.

Online algorithms are evaluated using a competitive analysis approach [11–13]. The competitive analysis measures the performance of an online algorithm against an optimum offline algorithm. It is assumed that the optimal offline algorithm has complete knowledge of the future inputs and always makes an optimum decision. Formally competitive analysis is defined as follows [12,13]. Let **ON** be an online algorithm, and **OPT** be an optimum offline algorithm for a cost minimization problem **P**. Let ***I*** be the set of all possible input instances for **P**. **ON**(I) and **OPT**(***I***) represents the cost incurred by **ON** and **OPT** on input instance $I \in \boldsymbol{I}$. **ON** is called c-competitive, if $\forall I \in \boldsymbol{I}$

$$\mathbf{ON}(I) \leq c.\mathbf{OPT}(I) + \alpha \tag{1}$$

where $\alpha \geq 0$ is a constant. This means that for all possible inputs, the cost of online algorithm **ON** will be no more than c times the cost of **OPT**.

One of the key advantages of using a competitive ratio as a performance measure is its ability to avoid dependence on input distribution [14,15]. Other measures such as average case analysis, rely heavily on the input distribution. Any deviation from the assumed distribution can render the results invalid. Competitive analysis, in contrast, does not assume any input distribution, and the input can

be drawn from any probability distribution. Therefore, the results are valid for all possible inputs. For more details on the working of competitive analysis, the reader is referred to the seminal work of [16].

In this work, we contribute towards the energy efficient cloud computing by.

- Designing optimal offline and online algorithms for the single host virtual machine migration problem to decide on an optimal time for virtual machine migration to reduce the overall energy cost of a data center
- Performing theoretical analysis of the proposed online algorithm using the competitive analysis technique
- Performing experimental evaluation of the proposed algorithm using a real-world dataset against the benchmark algorithms

The rest of the work is organized in the following manner. Section 2 presents formal problem settings. Section 3 succinctly reviews state-of-the-art literature. The optimum offline and online algorithms are presented in Section 4. Competitive analysis of the algorithms is performed in the same Section. Experimental settings including methodology, datasets and evaluation criteria are discussed in Section 5, whereas results and discussions are presented in Section 6. Section 7 concludes the work.

2 Formal Problem Settings

We consider a single host H with processing power M that can run at variable speed $h_j(j = 1, 2, \ldots, L)$. Note that these levels are discrete. Virtual machines are represented by $V_i\,(i = 1, 2, \ldots, N)$. Each V_i has a maximum processing requirement of M_i, which can change with the time but cannot exceed M, i.e., $\sup M_i \leq M \forall i \in [1, N]$. Current processor requirement of V_i at time t is represented by C_{it}. Cost of SLAV per unit time is C_v. V_s represents the start time of SLAV and V_f is the time when SLAV ends, i.e., the processing requirement of the current host decreases again after SLAV. Migration time for V_i at time t is a function of C_{it} and is modelled as $f(C_{it})$. Migration cost per unit time of V_i starting at time t is modelled as $g(V_{it})$. m is the time point when migration is initiated by an algorithm. Total cost includes migration cost and SLAV cost.

Without loss of generality, we assume that at each time point $t \in [1, T]$, either a new virtual machine V_i with a processing requirement C_{it} arrives or an existing virtual machine modifies its processing requirement. The host H is oversubscribed if the sum of all processing requirements at time t is more than the processing power M of the host, i.e., $\sum_{i=1}^{K} C_{it} > M$, where $K \leq N$is the total number of virtual machines assigned to host H at t. When the host is oversubscribed, Service Level Agreement Violation (SLAV) occurs, i.e., the host is unable to meet the processing demands of the existing virtual machines. At this stage, the algorithm has two choices. One, to provide an inferior level of quality of service and continue accruing SLAV cost. Two, to initiate a virtual machine migration by transferring a virtual machine to a new host and incur a migration cost. The decision has to be taken without the knowledge of future virtual machine arrivals and change in the processing requirements of the existing virtual machines. The research problem is to design an optimal algorithm that can decide if virtual machine migration shall be initiated when a host H is oversubscribed (resulting in extra energy cost of the new physical host), or to delay the instantiation of the new host and accrue SLAV cost.

3 Literature Review

In the following, we present a summary of approaches for VM migration that are proposed in the literature. Due to space constraints, only important works are reported.

One of the seminal works introducing online algorithms for the single host virtual machine migration is presented by Beloglazov et al. [2]. They introduced an online algorithm as described in Algorithm 1 (henceforth referred to as A_{BB}). The algorithm starts migration as soon as it encounters the start of SLAV, i.e., start migration at $m = V_s$. Authors derived a competitive ratio of $2 + s$ where $s \in R^+$. As s can have any positive value, therefore the competitive ratio is not in closed form.

Verma et al. [17] represented the application placement problem as a bin-packing problem. The servers were modelled as bins and VMs as balls. In their work, they focused on minimizing power and migration costs whilst maintaining the performance requirements. Kusic et al. [18] used a Limited Lookahead Control (LLC) for dynamic resource provisioning in a virtualized computing environment. They investigated the problem of SLA violation and minimizing energy utilization in virtualized data centres. In order to estimate the future requests a Kalman filter is used. One of the key drawbacks of these initial works is the lack of consideration for the quality of service aspect. Buyya et al. [1] presented heuristics for energy efficient resource allocation and management in cloud computing environments considering QoS requirements. Feller et al. [19] modelled the problem of dynamic workload placement as an instance of the multidimensional bin-packing problem. They introduced a novel dynamic workload placement algorithm based on the Ant Colony Optimization meta-heuristic to solve the above-mentioned problem. One of the key problems with heuristics is the lack of theoretical foundations. No theoretical performance guarantees are provided for heuristics.

Beloglazov et al. [20] proposed resource allocation algorithms by dynamic consolidation of VMs into a minimum number of servers. They aimed to minimize the SLA violations experienced due to the workload consolidation strategies. Their proposed algorithms are based on a modified Best Fit Decreasing approach. Kansal et al. [21] designed an algorithm for virtual machine migration using the Firefly optimization technique. The authors proposed to transfer a maximally loaded virtual machine to another physical machine. However, the authors did not consider the SLAV factor, thus it is possible that service degradation is faced by the end user, and no penalty is imposed on the service provider. Fu et al. [9] presented a novel approach for balancing the load of network resources in a data centre. The proposed algorithm divided the data centre into various regions based on the bandwidth consumption of the hosts. The processing load was then balanced across various regions by virtual machine migration. The overall objective was to obtain balanced network resource utilization across the data centre. However, the authors did not consider the impact on the quality of service. Further, no theoretical analysis of the proposed algorithm is conducted.

Shukla et al. [10] identified two main problems that commonly occur during virtual machine migrations. These problems are downtime and total migration time. Downtime is the time duration in which a VM is not available to the user, whereas total migration time is the time duration from the start of the downtime till the complete transfer of the virtual machine to a new physical host. The authors proposed a multi-phase approach for live migration in which the process of live migration is divided into various phases. In phase one, all the memory pages are migrated. In phase two, a decision about the migration based on the history of each page is taken. The third phase implements a forecasting mechanism to predict the behaviour of the page in the next time frame. Based on the predicted outcome a decision about the live migration is taken. Choudhary et al. [22] provided a detailed survey of the literature on virtual machine migration. The authors discussed various types of migrations and categorized the techniques based on duplication mechanism, and context awareness. Noshy et al. [23] considered the problem of live virtual machine migration and reviewed state of the art optimization techniques in the context of live virtual machine migration. The authors also provided directions for future research.

Karthikeyan et al. [24] argued that in a cloud data centre environment, virtual machine migration increases associated energy costs as well as the internal rate of failure. The authors proposed a Naive Bayes classifier with hybrid optimization using Artificial Bee Colony–Bat Algorithm to reduce the energy consumption in virtual machine migration. A critical drawback of the study is the lack of consideration for the Quality of Service parameter in the proposed scheme. In their work, Moghaddam et al. [25] focused on reducing the overall energy costs by minimising the probability of virtual machine migrations. The authors used a combination of Cellular Learning Automata based Evolutionary Computing (CLA-EC) and neuro-fuzzy techniques to reduce the possible numbers of VM migrations. The authors used workload traces from PlanetLab to show that the proposed technique reduces the number of VM migrations by 59%. In terms of energy minimization in cloud servers in relation to virtual machine migration, Hieu et al. [26] adopted a unique approach. The authors argued that heavy reliance on virtual machine migration increases the overhead and thus energy wastage. Hieu et al. [26] employed multiple usage prediction which considers the historical usage as well as the predicted future usage of multiple resources (such as CPU, bandwidth, RAM usage etc.) for a more reliable characterization of underload and overload servers.

For a detailed survey highlighting the various aspects of live migration in the cloud computing environment, the reader is referred to Le [27]. For energy efficiency in geographically distributed data centres, the reader is referred to Ahmad et al. [3].

From the literature review, we identified that the current approaches mostly rely on optimization techniques which results in heuristics approaches. These approaches, though providing good performance on data sets, can suffer from data snooping bias. In data snooping bias, an algorithm designed based on specific data set, may not actually perform well on other data sets. In addition, no theoretical guarantees (bounds) with respect to the performance of the algorithm are presented. In our work, we aim to bridge this gap by considering service level agreement and designing an optimal online algorithm with guaranteed worst-case performance.

4 The Proposed Algorithm

In this section, we present an optimum offline algorithm A_{OPT} and an optimum online algorithm A_{ON} for the single host virtual machine migration problem. Further, we analyze A_{ON} using the competitive analysis approach.

4.1 Optimum Offline Algorithm

An optimum offline algorithm has complete knowledge about the input sequence as well as about the start and end timings of SLAV. Therefore, it has the ability to make the best possible decision. Refer to Algorithm 2 for the optimum offline algorithm.

It is clear from Algorithm 2 that the optimum offline algorithm will only initiate the migration of a virtual machine when the SLAV cost is exceeding the migration cost, otherwise, it is more beneficial accruing the SLAV cost only. Recall that an offline algorithm has complete knowledge of the input sequence, therefore, it is aware of the SLAV occurrence, and the start and end time of SLAV.

4.2 Optimum Online Algorithm

Though optimum offline algorithm achieves the best possible performance (minimum cost) for any given input, in the real world it is not possible to have advanced knowledge of SLAV and VM requirements. This necessitates the need for an online algorithm that can achieve the best possible

result on unforeseen input sequences, i.e., the algorithm needs to decide if and when it needs to start a migration?

It is possible that the online algorithm starts migration too early and later she observes that it would have been better not to migrate at all (referred to as *too early error*). Likewise, it is also possible that the online algorithm starts the migration too late, and in the hindsight, it would have been better to initiate a migration as soon as the SLAV occurred (referred to as *too late error*). Therefore, the online algorithm has to find a time point to balance the *too early error* and the *too late error*. Let us assume that W_t is the time that an online algorithm will wait after the start of SLAV before initiating a migration. Therefore, the best decision in this situation is to start migration only when the SLAV cost equals migration cost, i.e.,

$$SLAVCost = MigrationCost$$

$$W_t.C_v = f\,(C_{it})\,.g\,(V_{it})$$

$$W_t = \frac{f(C_{it}).g(V_{it})}{C_v} \tag{2}$$

When SLAV starts at V_s, online algorithm will calculate a waiting time W_t and will initiate a migration at time $m = V_s + W_t$ only if the SLAV still continues at $V_s + W_t$. The waiting time W_t is the threshold at which the migration and SLAV costs equates. The algorithm is presented in Algorithm 3.

Theorem 1: Algorithm 3 (A_{ON}) achieves a competitive ratio of $2 + \frac{C_v}{g(V_{it})}$.

Proof: Consider a worst-case sequence as shown in Fig. 2. Assume that the SLAV begins at time V_s and lasts till $V_s + f(C_{it})$. We bound the cost of A_{OPT} and our proposed online algorithm A_{ON} as follows;

Figure 2: The worst-case input sequence

A_{OPT} has complete knowledge of the input sequence and knows that the SLAV that begins at V_s will last till $V_s + W_t + f(C_{it})$. The best possible decision to reduce the over all cost is to initiate a migration $V_s + f(C_{it})$ and avoid the SLAV cost. The only cost incurred by A_{OPT} is the migration cost which is obtain by multiplying migration cost per unit time with total migration time as shown in the following equation.

$$Total\ Cost\ of\ A_{OPT} = Migration\ Cost\ of\ A_{OPT} = f\,(C_{it})\,.g\,(V_{it})$$

Our proposed optimum online algorithm (A_{ON}) will wait till time $V_s + W_t$ before initiating a migration. It starts migration at $V_s + W_t$ and will complete the migration at $V_s + W_t + f(C_{it})$. This is also the time when the SLAV ends. The cost of A_{ON} includes both SLAV and migration cost calculated as follows.

$$Total\ Cost\ of\ A_{ON} = C_v\,(W_t + f\,(C_{it})) + f\,(C_{it})\,.g\,(V_{it})$$

Therefore, the competitive ratio c_{ON} achieved by A_{ON} is;

$$c_{ON} = \frac{Total\ Cost\ of\ A_{ON}}{Total\ Cost\ of\ A_{OPT}}$$

$$c_{ON} = \frac{C_v\left(W_t + f\left(C_{it}\right)\right) + f\left(C_{it}\right).g\left(V_{it}\right)}{f\left(C_{it}\right).g\left(V_{it}\right)}$$

$$c_{ON} = 2 + \frac{C_v}{g(V_{it})} \tag{3}$$

Corollary 1: As the cost of SLAV per unit time (C_v) is less than the migration cost per unit time $g(V_{it})$, $c_{ON} < 3$.

Corollary 1 confirms the better performance achieved by our proposed algorithm A_{ON} in relation with A_{BB}.

5 Experimental Setting

5.1 Methodology

We evaluate our proposed algorithm (A_{ON}) by comparing its performance against the benchmark algorithms A_{BB} and A_R (see Section 5.3). For this purpose, we used CloudSim - a well-established open-source simulator implemented in Java [28], which has been extensively used previously to evaluate the performance of new VM migration algorithms [25]. CloudSim offers implementations of a wide range of popular VM migration algorithms which can be modified to devise new variations. Moreover, a popular real-world dataset, Planetlab workloads [2], is already integrated with CloudSim, which makes it easier to set up a reproducible experimental environment.

To define a migration algorithm fully in CloudSim, two policies need to be specified; a VM allocation policy to optimize the allocation of VMs according to current utilization levels of various hosts, and a VM selection policy to select VMs to be migrated from over-utilized hosts to under-utilized hosts. We use median absolute deviation (MAD) as A_{BB} VM allocation policy [2], which is already implemented in CloudSim. As its name suggests, this policy computes the median absolute deviation of CPU utilization history for each host to determine whether it is over-utilized or not. Moreover, the computation of an optimal mapping between over-utilized and under-utilized hosts for migrating VMs from the former to the latter also comes under this policy. We use minimum migration time (MMT) as VM selection policy in A_{BB}, which is also already implemented in CloudSim. This policy returns the smallest VM, in terms of its memory requirement, for migration from an over-utilized host.

As explained, A_{ON} does not initiate a VM migration immediately from over-utilized hosts. Instead, it calculates a waiting time (W_t) before proceeding with the VM migration. To implement this delay mechanism, we made extensive modifications to various files related to the implementation of MAD. However, the implementation of MMT remained unchanged.

In our modified implementation of MAD, we compute W_t periodically for each over-utilized host with a scheduling interval (S_t) which is specified in CloudSim before starting a simulation. The value of W_t turns out to be finite and positive i.e., a meaningful delay has been computed only if its corresponding value of SLAV cost (C_v) is greater than zero. In case the value of C_v is less than or equal to zero, we do not delay VM migration from the host, and proceed according to the original allocation policy. Upon the computation of a finite and positive value of W_t for a host, we skip to the next host

without initiating VM migration. Since this process occurs only at periodic intervals of S_t, we cannot delay VM migration from a host by an exact amount of W_t if it is not an exact multiple of W_t. In practice, the actual delay (W_t) depends on S_t as shown in the following equation.

$$\overline{W_t} = \left\lceil \frac{W_t}{S_t} \right\rceil S_t$$

It is evident that the minimum value of $\overline{W_t} = W_t$ i.e., VM migration from the current host is delayed by a time period of at least S_t. The default value of S_t in CloudSim is 300 s (5 min). This is same as the interval at which CPU utilization readings were taken for all the ten Planetlab workloads used in our experiments.

We compared the performance of A_{ON} against A_{BB} and A_R for all the ten Planetlab workloads. During our experiments, we did not modify default values of different parameters already initialized in CloudSim. A summary of these parameters is provided in Tab. 1. The set of 800 hosts used in each experiment consisted of 400 hosts of each type shown in Tab. 1. Similarly, the total number of VMs during each experiment was equally distributed among the four available VM types shown in Tab. 1.

Table 1: Default values of different parameters defined in CloudSim

Parameters	Values
Scheduling interval (S_t)	300 s
Total simulation duration for each workload	24 h
Total number of hosts	800
Specifications of host type 1	HP ProLiant ML110 G4: Xeon 3040 1860 MHz CPU with 2 cores, 4 GB RAM
Specifications of host type 2	HP ProLiant ML110 G5: Xeon 3075 2660 MHz CPU with 2 cores], 4 GB RAM
Types of VMs	4
Specifications of VM type 1	2500 MIPS, 870 MB RAM, 2500 MB VM size
Specifications of VM type 2	2000 MIPS, 1740 MB RAM, 2500 MB VM size
Specifications of VM type 3	1000 MIPS, 1740 MB RAM, 2500 MB VM size
Specifications of VM type 4	500 MIPS, 613 MB RAM, 2500 MB VM size

After each experiment, we recorded values for total energy consumption (E_{total}), number of VM migrations (N_{vm}), overall SLA violation ($SLA_{overall}$) and average SLA violation (SLA_{avg}). We did not observe any variation in the values of these attributes over multiple runs of our experiments.

5.2 Datasets

In order to ensure a fair comparison, we used the same data as used by [2]. The data is part of CoMon project which is a monitoring infrastructure for PlanetLab [2]. It contains CPU utilization of more than thousands of virtual machines measured from servers located at different places all around the world. The data of ten random days are chosen from the workload traces that are collected in March and April 2011. The data files contain CPU utilization values of virtual machines measured every 5 min for 24 h. Note that each line in a file represents a single request, and data for each day are combined in a single folder. In total, all the folders contain 11,746 files, which contain over 3.3 million user requests. The properties of the data are summarized in Tab. 2.

Table 2: Dataset statistics

Date	Number of files	Number of requests	Mean (%)	Std deviation (%)
03-Mar-11	1052	302976	12.31	17.09
06-Mar-11	898	258624	11.44	16.83
09-Mar-11	1061	305568	10.7	15.57
22-Mar-11	1516	436608	9.26	12.78
25-Mar-11	1078	310464	10.56	14.14
03-Apr-11	1463	421344	12.39	16.55
09-Apr-11	1358	391104	11.12	15.09
11-Apr-11	1233	355104	11.56	15.07
12-Apr-11	1054	303552	11.54	15.15
20-Apr-11	1033	297504	10.43	15.21

5.3 Benchmark Algorithms

We compare the performance of our proposed algorithm (A_{ON}) with two benchmark algorithms A_{BB} and A_R. Note that A_R is a randomized algorithm and is defined in Algorithm 4. Algorithm 4 is a randomized algorithm which with 0.5 probability executes Algorithm 1, otherwise calls Algorithm 3.

We do not compare our proposed algorithm with algorithms based on heuristic techniques as such techniques are computationally more expensive than our proposed online algorithm. Note that our proposed online algorithm A_{ON} has a worst-case time complexity of $O(1)$.

6 Results and Discussions

We use three metrics to compare the performance of A_{ON}, A_{BB} and A_R, namely total energy consumption, number of VM migrations, percentage of overall SLA violations.

In terms of total energy consumption, A_{ON} is observed to be the best algorithm. A_{ON} consumed 25% less energy than A_{BB} and 10.6% less energy than A_R. Fig. 3 summarizes the daily energy consumed by each of the algorithms on the dataset. It can be observed that on all of the ten days of the dataset, the energy consumed by the proposed algorithm A_{ON} is less than A_R and A_{BB}. We also investigated the consistency in term of energy consumed by the algorithms by measuring the standard deviation over the ten days period. We observed that our proposed algorithm A_{ON} achieved the minimum standard deviation of the three algorithms, whereas A_{BB} recorded the highest standard deviation, signifying that A_{ON} is consistently outperforming A_{BB} in term of energy consumption.

Another measure that we used to evaluate the performance of algorithms is the number of virtual machine migrations carried out by each of the three algorithms. Recording the number of VM migrations is important as it can help explain the total energy consumption. Intuitively, the larger the number of migrations, the higher the total energy consumed. In our experiments, we observed that A_{BB} performed a total of 263,051 migrations over the ten days period, whereas the corresponding number of migrations for A_R and A_{ON} are 187,390 and 147,672 respectively. Our proposed algorithm performed 43.8% less migrations than A_{BB}, and 21% less migrations than A_R. Fig. 4 summarizes the number of migrations performed by the algorithms.

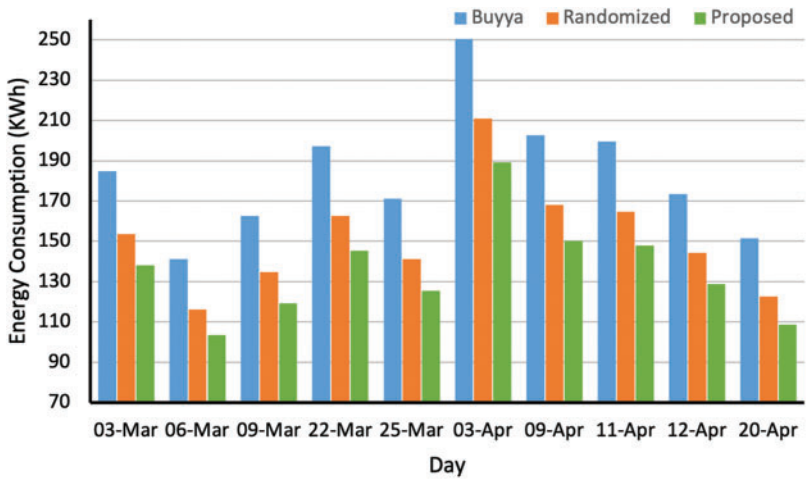

Figure 3: Daily energy consumption by A_{ON}, A_{BB} and A_R

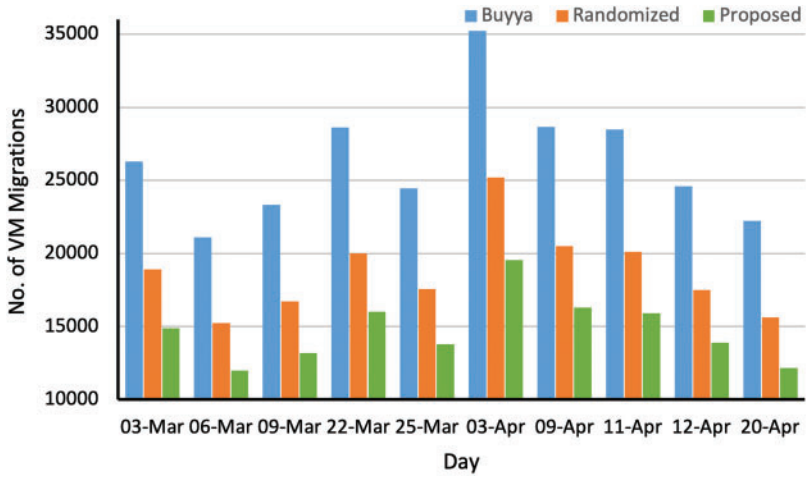

Figure 4: Number of VM migrations by A_{ON}, A_{BB} and A_R

We also measured the SLAV for the three algorithms. A_{BB} observed the least number of SLA violations. On average, A_{BB} observed SLA violations in only 0.08% of the all the requests. A_R observed SLA violations in 0.7% of all the requests, whereas the corresponding value for A_{ON} is 1.35%. The percentage of SLA violations for A_{BB} is extremely low as it initiates migration as soon as the SLAV occurs, whereas for A_{ON}, the percentage of SLAV violations are higher as the algorithm initiates migration only after waiting for a certain period of time W_t. This feature of the A_{ON} might increase SLAV cost but reduces the overall cost of the data center which is the ultimate objective.

7 Conclusion

Energy efficiency is a key contemporary requirement for data centers both from the monetary and environmental perspectives. In an attempt to minimize energy usage, data centers use a variety of techniques including virtual machine migration. In this work, we proposed a novel algorithm for single host virtual machine migration problem. We analyzed our proposed algorithm both theoretically using

the worst-case competitive ratio as a performance measure, and experimentally using a real-world dataset from planet-lab. Our proposed algorithm outperforms the standard benchmark algorithms from the literature and achieves a performance improvement of 26% in terms of total energy consumption, whereas in terms of number of migrations the improvement is over 43%. Likewise, the proposed algorithm is more consistent than the standard benchmark algorithms too.

The work can be extended to design randomized algorithms for single host virtual machine migration problem. Further, the proposed approach can be extended to design energy efficient algorithms for multi-host virtual machine migration and consolidation. It will be of interest to design a machine learning framework to predict the future requests/demands of virtual machines.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg and I. Brandic, "Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer Systems*, vol. 25, no. 6, pp. 599–616, 2009.

[2] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers," *Concurrency and Computation: Practice and Experience*, vol. 24, no. 13, pp. 1397–1420, 2012.

[3] I. Ahmad, M. I. K. Khalil and S. A. A. Shah, "Optimization-based workload distribution in geographically distributed data centers: A survey," *International Journal of Communication Systems*, vol. 33, no. 12, pp. e4453, 2020.

[4] C. Belady, "In the data center, power and cooling costs more than the IT equipment it supports 2007," 2007. [Online]. Available: http://www.electronics-cooling.com/articles/2007/feb/a3.

[5] L. A. Barroso and U. Holzle, "The case for energy-proportional computing," *Computer*, vol. 40, no. 12, pp. 33–37, 2007.

[6] R. Huang and E. Masanet, Data center IT efficiency measures. In: *Nat. Renew. Energy Lab. (NREL)*. Golden, CO, USA, National Renewable Energy Laboratory, 2015.Rep. NREL/SR–7A40- 63181.

[7] R. M. Fujimoto, M. Hunter, A. Biswas, M. Jackson and S. Neal, "Power efficient distributed simulation," in *Proc. of the 2017 ACM SIGSIM Conf. on Principles of Advanced Discrete Simulation*, Singapore, Singapore, pp. 77–88, 2017.

[8] A. Shehabi, S. Smith, D. Sartor, R. Brown, M. Herrlin *et al.,* United states data center energy usage report. In: *Technical Report*. Berkeley, CA (United States): Lawrence Berkeley National Lab. (LBNL), 2016.

[9] X. Fu, J. Chen, S. Deng, J. Wang and L. Zhang, "Layered virtual machine migration algorithm for network resource balancing in cloud computing," *Frontiers of Computer Science*, vol. 12, no. 1, pp. 75–85, 2018.

[10] R. Shukla, R. K. Gupta and R. Kashyap, "A multiphase pre-copy strategy for the virtual machine migration in cloud," in *Smart Intelligent Computing and Applications*. Singapore: Springer, pp. 437–446, 2019.

[11] E. Mohr, I. Ahmad and G. Schmidt, "Online algorithms for conversion problems: A survey," *Surveys in Operations Research and Management Science*, vol. 19, no. 2, pp. 87–104, 2014.

[12] J. Iqbal and I. Ahmad, "Optimal online k-min search," *EURO Journal on Computational Optimization*, vol. 3, no. 2, pp. 147–160, 2015.

[13] J. Iqbal, I. Ahmad and A. Shah, "Competitive algorithms for online conversion problem with interrelated prices," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 6, pp. 582–589, 2019.

[14] H. Kaplan, D. Naori and D. Raz, "Competitive analysis with a sample and the secretary problem," in *Proc. of the Fourteenth Annual ACMSIAM Symp. on Discrete Algorithms*, Salk Lake City, USA, SIAM, pp. 2082–2095, 2020.
[15] J. Iqbal, I. Ahmad and A. Shah, "Disparity between theory & practice: Beyond the worst-case competitive analysis," in *2018 IEEE 5th Int. Conf. on Engineering Technologies and Applied Sciences (ICETAS)*, Bangkok, Thailand, IEEE, pp. 1–6, 2018.
[16] R. M. Karp, "On-line algorithms versus off-line algorithms: How much is it worth to know the future?," *IFIP Congress*, vol. 12, pp. 416–429, 1992.
[17] A. Verma, P. Ahuja and A. Neogi, "pMapper: power and migration cost aware application placement in virtualized systems," in *Proc. of the 9th ACM/IFIP/USENIX Int. Conf. on Middleware*, New York, Springer-Verlag, pp. 243–264, 2008.
[18] D. Kusic, J. O. Kephart, J. E. Hanson, N. Kandasamy and G. Jiang, "Power and performance management of virtualized computing environments via lookahead control," *Cluster Computing*, vol. 12, no. 1, pp. 1–15, 2009.
[19] E. Feller, L. Rilling and C. Morin, "Energy-aware ant colony based workload placement in clouds," in *Proc. of the 2011 IEEE/ACM 12th Int. Conf. on Grid Computing*, Lyon, France, IEEE Computer Society, pp. 26–33, 2011.
[20] A. Beloglazov, J. Abawajy and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," *Future Generation Computer Systems*, vol. 28, no. 5, pp. 755–768, 2012.
[21] N. J. Kansal and I. Chana, "Energy-aware virtual machine migration for cloud computing-a firefly optimization approach," *Journal of Grid Computing*, vol. 14, no. 2, pp. 327–345, 2016.
[22] A. Choudhary, M. C. Govil, G. Singh, L. K. Awasthi, E. S. Pilli *et al.,* "A critical survey of live virtual machine migration techniques," *Journal of Cloud Computing*, vol. 6, no. 1, pp. 23, 2017.
[23] M. Noshy, A. Ibrahim and H. A. Ali, "Optimization of live virtual machine migration in cloud computing: A survey and future directions," *Journal of Network and Computer Applications*, vol. 110, no. 2, pp. 1–10, 2018.
[24] K. Karthikeyan, R. Sunder, K. Shankar, S. Lakshmanaprabu, V. Vijayakumar *et al.,* "Energy consumption analysis of virtual machine migration in cloud using hybrid swarm optimization (abc-ba)," *The Journal of Supercomputing*, vol. 76, no. 5, pp. 3374–3390, 2020.
[25] M. J. Moghaddam, A. Esmaeilzadeh, M. Ghavipour and A. K. Zadeh, "Minimizing virtual machine migration probability in cloud computing environments," *Cluster Computing*, vol. 23, no. 4, pp. 3029–3038, 2020.
[26] N. T. Hieu, M. Di Francesco and A. Yla-Jaaski, "Virtual machine consolidation with multiple usage prediction for energy-efficient cloud data centers," *IEEE Transactions on Services Computing*, vol. 13, no. 1, pp. 186–199, 2020.
[27] T. Le, "A survey of live virtual machine migration techniques," *Computer Science Review*, vol. 38, no. 8, pp. 100304, 2020.
[28] T. C. Computing and M. U.O., "Distributed systems laboratory," Cloudsim, 2020. [Online]. Available: http://www.cloudbus.org/cloudsim/.