CoLocateMe: Aggregation-Based, Energy, Performance and Cost Aware VM Placement and Consolidation in Heterogeneous IaaS Clouds

Muhammad Zakarya®, Lee Gillam®, Khaled Salah®, Omer Rana®, Santosh Tirunagari®, and Rajkumar Buyya®

Abstract—In many production clouds, with the notable exception of Google, aggregation-based VM placement policies are used to provision datacenter resources energy and performance efficiently. However, if VMs with similar workloads are placed onto the same machines, they might suffer from contention, particularly, if they are competing for similar resources. High levels of resource contention may degrade VMs performance, and, therefore, could potentially increase users’ costs and infrastructure’s energy consumption. Furthermore, aggregation-based methods result in stranded resources and, therefore, less economics. The recent industrial interest in segregating workloads opens new directions for research. In this article, we demonstrate how aggregation and segregation-based VM placement policies lead to variations in energy efficiency, workload performance, and users’ costs. We, then, propose various approaches to aggregation-based placement and migration. We investigate through a number of experiments, using Microsoft Azure and Google’s workload traces for more than twelve thousand hosts and a million VMs, the impact of placement decisions on energy, performance, and costs. Our extensive simulations and empirical evaluation demonstrate that, for certain workloads, aggregation-based allocation and consolidation is ~9.61% more energy and ~20.0% more performance efficient than segregation-based policies. Moreover, various aggregation metrics, such as runtimes and workload types, offer variations in energy consumption and performance, therefore, users’ costs.

Index Terms—Clouds, datacenters, VM placement, resource consolidation, migrations, heterogeneity, energy efficiency, performance

1 INTRODUCTION

One of the major challenges in cloud datacenters is to manage computational resource energy and performance efficiently. Energy consumption affects our environment and account for large energy bills while performance affects cloud economics. Therefore, cloud service providers are focusing to design policies for energy, performance aware computing, encouraged by high operational costs of installed computer clusters [1]. The goal can be achieved in two different ways: (i) assigning only appropriate resources; and (ii) consolidating workload onto fewer machines using VM migration and switching off idle machines. On one side, the capability of VM migrations brings several benefits such as improved manageability, increased utilisation and energy savings. However, on the other side, it results in down time that decreases the performance of workloads. Migrations are expensive and in dynamic cloud environments, where thousands number of VM requests arrive in an hour, even they might not be suitable. Therefore, appropriate VM placement policies are essential to save energy and provide customers the expected level of workload performance [2].

VM placement policies can be categorized as: (a) segregation; and (b) aggregation based [1]. In segregation based policies, the providers run user-facing, batch, and production workloads in separate clusters (hosts). Therefore, if workloads demand is either low or even not available, then their resources still need to be switched on and this may also result in stranded resources. Large number of hosts in use can increase the providers’ energy bill and have impact on our environment. Aggregation based policies run mixed workloads on same hosts which may degrade the workload performance, particularly, if they compete for similar resources (co-located VMs) [3]. Moreover, workload performance also varies across different CPU architectures – similar workloads may run quite differently over same CPU model [4]. Subsequently, lower workload performance could potentially increase infrastructure energy consumption and users monetary costs. The former approach is widely used in many production clouds, such as Alibaba cluster [5], with the notable
exception of the Google’s cluster [6]. Perhaps, inspired from benefits of aggregation-based approaches, Alibaba’s cluster resources are also now offered to run workloads in mix. However, a detailed investigation of both methodologies is still needed in terms of energy efficiency and workload performance. In fact, the providers’ switching efforts motivate us to investigate which technique is better than the other in term of energy, performance, and cost efficiency. Furthermore, which characteristics of the workloads should be used to aggregate them onto similar servers.

In this paper, we investigate how VMs and workloads would be placed onto physical hosts, in a heterogeneous cluster, so that the infrastructure energy consumption is minimised under the performance and users’ cost constraints. We propose runtime-aware aggregation-based, energy, performance, cost (EPC) efficient VM placement and consolidation policies in order to execute several workloads in mix. Since, workloads are co-located, therefore, we call it CoLocateMe. Using real workload datasets from virtualised clouds, such as Google and Microsoft Azure clouds, we evaluate the performance of runtime-aware aggregation and segregation based placement policies, in an event driven cloud simulator i.e., CloudSim [7]. Our empirical evaluation suggests that the proposed, runtime-aware aggregation-based, VM placement and consolidation policies outperform segregation-based policies. Since, aggregation favours to collocate VMs having similar characteristics onto similar hosts, therefore, we call it “CoLocateMe”. Major contributions of this paper are:

- an aggregation-based, energy, performance and cost (EPC) aware VM placement policy is proposed;
- a consolidation method is suggested that put similar workloads onto same resources;
- with respect to workload performance, we model resource heterogeneities in datacenters; and
- we evaluate the impact of aggregation and segregation-based VM placement and migration policies on infrastructure energy efficiency, workload performance and users’ costs.

The rest of the paper is organized as follows. In Section 2, we discuss the VM placement problem. In Section 3, we propose an aggregation-based allocation and consolidation technique that places similar workloads on same resources. We validate the proposed scheme using real workload traces from Azure clusters in Section 4. We offer an overview of the related work in Section 5. Finally, Section 6 concludes the paper and describes future research.

2 PROBLEM DESCRIPTION

The runtime period or execution time of a VM ($R_{vm}$) is dependent on data size to be processed and the quantity of resources i.e., CPU cores, memory, and bandwidth, assigned. The active period of a physical machine is proportional to the longest runtime of the VMs running on the machine. If the duration of most VMs is much shorter than the runtime of the longest one, it indicates low machine runtime efficiency. To increase the runtime efficiency of machines, researchers have proposed techniques like aggregating VMs with similar runtime to a particular cluster or consolidate VMs by their capacities [8]. The former method can save more power than the later one through decreasing machine runtime. However, the impact of the runtime diversity of VMs and VM resource capacities on amount of machines should be considered when designing energy and performance efficient resource management policies. Furthermore, performance loss due to resource contention must be taken into account.

Considering only two hosts, as part of a datacenter, as shown in Fig. 1, each one can accommodate two VMs. There are four VMs with different runtime requirements. If the placement is not runtime aware and consolidation is not assumed; then, host A will at least run for 10 hours and host B for 14 hours (right-hand side i.e., VM-1 and VM-2 are co-located on host A while VM-3 and VM-4 on host B), continuously. However, if runtimes are considered (left-hand side – VM-2 and VM-4 are co-located on host A while VM-1 and VM-3 on host B) then after 2 hours, host A can be switched off to save power; if there are no pending VM requests in the admission queue. Note that, the runtime of each VM depends on the application’s workload which is highly unpredictable. If we can predict it, this does not essentially means that hosts in Fig. 1 (right) will run for 10 and 14 hours. When VM-2 and VM-4 finish their jobs, we can consolidate VM-1 and VM-3 to one host. Similar decisions can also be taken on VM capacities and instances of similar types can be placed together. However, various workloads if co-located or aggregated may not perform up to expected levels. However, if workloads compete for similar resources, then performance of the workloads may potentially be affected [3]. Furthermore, similar workloads may perform quite differently across same CPU models due to: (i) CPU models [4]; and (ii) resource contention [3]. These variations may potentially affect workload runtimes, therefore, users’ service costs; and energy consumption. Subsequently, energy consumption affects revenue of service providers (energy bills) and our environment (green house gases). Therefore, it is essential to account for these costs when deciding resource placement and consolidation.

The above problem can be formulated as a multi-objective optimisation problem where objective are: (a) minimise total energy consumption ($E = \Sigma_{i=1}^{N_r} E_{host_i}$); improve or, at least, maintain workload performance ($P$); and minimise users’ monetary costs ($C$ i.e., VMs provisioning costs). Note that, $P$ and $C$ are inversely proportional where improving workload’ $P$ means reducing its $R$ (the sum of all VM runtimes $R_{vm}$ running the same workload) which subsequently means reducing $C$ (the sum of all VMs provisioning costs).

Fig. 1. Problem description [aggregation w.r.t runtimes].
For many request-response cloud workloads such as databases, with the notable exception of batch workloads, this statement might not be essentially true due to users’ experiences. Furthermore, improving \( P \) through provisioning more resources will itself increase certain costs (e.g., \( C, E \)). For any request-response workloads such as web services, streaming analytics, getting a response faster does not reduce the energy cost, and would most likely increase it. Therefore, our aim is to improve \( P \) through running workloads on appropriate VMs and hosts (based on historical information). Moreover, various objectives (\( E, P, C \)) can be combined into a single metric (\( ERC = \frac{E_{PS}}{P} \)) where \( P \) is the inverse of runtime (\( R \)), and then solved as a single objective problem [1], given by

\[
\min(ERC).
\]

As, energy is measured in Wh (watt hours), runtime in hours and users’ monetary cost in $/hour. Thus, the above metric captures power to runtime ratio per unit cost [4]. The least values for \( ERC \) will translate to the best achievable performance & energy efficiency at the lowest cost. To deal with the above multi-objective optimisation problem, when one objective is preferred over another; then, the \( ERC \) can be elaborated as \( ERC = a \cdot E \times \beta \cdot R \times \gamma \cdot C \) subject to \( a + \beta + \gamma \in \{0, \ldots, 1\} \) (where \( a, \beta, \) and \( \gamma \) are the domination values for energy, performance, and user’s costs, respectively) [9], [10].

3 PROPOSED SOLUTION

Aggregation, based on runtime of VMs might be useful if workload runtimes are predictable. Albeit, various machine learning based techniques, such as gradient boosted trees [11], have been suggested to predict VMs runtimes. However, due to the unpredictable nature of the cloud workloads, many efforts are needed. In Section 3.1, we explain the methodologies and approaches to implement resource management policies. In Section 3.2, we explain the aggregation-based placement and consolidation policies.

3.1 Implementation Methodologies

Hence, due to the efforts involved in accurately predicting the runtimes of VMs; we use two different methodologies to implement the above algorithms: (i) use previous runtimes of VMs to aggregate them; and (ii) predict VMs runtimes using the gradient boost tree method [11]. However, there would be other efficient ways of doing the same, for example using workload types, VM sizes, as described later in Section 4.3.9.

3.1.1 Past Runtimes

From IaaS point of view, workloads and their runtimes cannot be predicted accurately. Therefore, instead of using workload actual runtimes, an alternative approach is to use their past runtimes (the duration for which the workload has already run). In other words, we assume that workloads which have run (in past) for similar runtimes may probably run (in future) for similar durations – this is like a probabilistic approach. The idea is based on our previous works [1], [12]; that use VMs or containers past runtimes in workload placement decisions – *migrate only relatively long-running VMs and/or containers since they could recover their migration costs*. We do not use the model here, instead, we use the methodology of migrating relatively log running VMs. Furthermore, the initial placement (i.e., past runtime is zero) is achieved through the classic first fit (FF) heuristic algorithm.

Using past runtimes for such decisions avoid complex prediction techniques (e.g., machine learning) that might not be reasonable in hyper-scale IaaS clouds.

3.1.2 Runtimes Prediction

If we assume that clouds are not opaque; then it is possible to predict their runtimes using historical data [13], [14]. Using past runtimes may not provide accurate estimates – for example, workloads which have run for longer durations have more probability and, therefore, higher tendency toward terminations. Therefore, it is essential to predict runtimes and use them in resource allocation and migration decisions. Note that, predicting a particular workload runtime may need identifying its type first i.e., CPU, memory, disk intensive. We assume that each VM requests certain resources (CPU, memory, storage), holds a priority and is initiated by a particular user. Moreover, the actual resource usage of each VM is also monitored. Since, submitting user, resource demand and actual usage have shown strong relationship to runtimes [11], [13]; therefore, we also used these features of more than ten millions tasks (categorised in three different groups w.r.t scheduling constants), using the Google’s cluster dataset, to train our prediction model. We used simple (linear regression) to complex (boosted trees) machine learning algorithms to estimate workload runtimes. Moreover, various techniques offer various levels of accuracy and, therefore, variations in experimental outcomes. Moreover, accurate predictions decrease the likelihood of inappropriate migration decisions. Similarly, predicting the runtime of VMs is also influenced by the type of workload hosted in the VM. In [15], the authors describe various approaches to accurate historical data if workloads differ. Further details on workload predictions can be found in [11], [14], [16], that offer reasonable accuracy for public clouds where workloads fluctuate more than private ones, significantly.

3.1.3 Migration Durations Prediction

When consolidating short running workloads, it is possible that the migration efforts are being wasted if the VM terminates during migration or just after its migration process is finished [1]. However, if preference is given to migrate VMs that would highly likely run for a long time after migration, the actual migration time would be a very small fraction of the VMs total lifetime. Further, the probability of the VM powering off during the migration time would be equally minuscule. To decide effective migrations, it is also essential to estimate the migration durations for VMs running different services. In [16], the authors have trained a machine learning approach to predict migration durations and other metrics using real workload dataset. Their investigations suggest that migration durations and performance degradation are, largely, reliant on the migration approach (such as pre-copy, post-copy) and workload type. Moreover, there is a strong linear relationship among the amount of data being copied and migration durations [17]. In off-line migration,
the performance loss (downtime) is almost equal to migration duration. However, in live migration, downtime is different and, usually, smaller than the migration duration [1]. In addition, the memory size of the VM is not always the best indicator on the performance drop when migrating. In many cases it is the rate of change in the dirty pages [17], [18].

In total migration time, page dirty rate plays an important role. To predict migration duration, it is essential that a representative workload is available to train various predictive models. Further, neither Google dataset [19] nor Microsoft Azure dataset [11] contain migration statistics of VMs. Therefore, it is difficult to estimate migration durations for tasks relating to these two datasets. Fortunately, an interesting VM migration dataset is presented in [16]. Therefore, we choose comparable workloads from Google, Microsoft Azure, and the migration datasets provided in [16]. This gives us simplified assumptions for comparing workload benchmarks and, therefore, estimation of accurate migration durations. The model was then trained using various approaches such as linear regression and support vector regression (SVR). Various features, such as VM size, page dirty rate, resource utilisation of VMs, source and destination servers, are considered. We have spent considerable efforts on statistical mapping of various workloads so that plausible assumptions can be derived for simulation purposes. Further details on mapping the Google’s workload traces [19] to real IaaS performance benchmarks [4] can be found in our previous works [1], [12].

### 3.2 Placement and Consolidation

In this section, we explain how the placement and consolidation techniques are being used in the optimisation module. The proposed policies ensure that the most runtime efficient hosts are selected to run particular VMs. The energy, performance aware placement policy, based on first fit technique (EPFF), is described in Algorithm 1. The energy, performance aware migration approach (EPAM) is described in Algorithm 2. The proposed aggregation-based placement is dependent on the runtime efficiencies of the hosts and VMs. First, the available hosts are being divided into groups based on their runtime efficiencies and temporal slack (as described in Section 3.2.1). Furthermore, all hosts in each group are being sorted in increasing order of their ERC values – the objective function (Eq. (1)). Then, based on the runtime of the VM (either predicted or past), using a particular classification technique (such as k-Means clustering algorithm as discussed in Section 3.2.2), every VM is categorised and mapped onto each host in a particular group. In fact, the latter step ensures that VMs of similar characteristics (run-times, capacities, etc.) are placed onto similar hosts i.e., aggregation (CoLocateMe). In case of segregation, this step is ignored from Algorithm 1 to ensure that VMs are not classified. As a result, the most runtime efficient host (i.e., at top of the list) is selected to run that particular group of VMs. Note that, the placement algorithm is a substantially improved version of the first fit (FF) heuristic approach.

The consolidation policy EPAM runs, periodically, every five minutes interval and looks for possibilities to aggregate and consolidate the workload (VMs) onto fewer hosts. Note that, a shorter time interval will lead to additional overhead but a larger one may lead to poor system performance as reacting to the dynamicity of the workload will be too late. Furthermore, this could happen in two different ways: (i) through migrating VMs from underloaded and overloaded hosts, using some pre-defined threshold values in terms of their utilisation levels (e.g., an upper threshold value $U_{\text{upper}}$ for overloaded hosts and a lower threshold value $U_{\text{lower}}$ for underloaded hosts) [1]; and (ii) through migrating all VMs from hosts having the highest levels of runtime efficiencies. Once hosts are being identified, a list of migratable VMs is constructed using a particular VM selection policy, using Algorithm 3. Several metrics of the VMs are considered when deciding their migrations. For example, [20] chooses a VM that either: (a) has a small memory so that its migration can be completed quickly; or (b) has maximum utilisation level so that overloaded host is avoided up to maximum. However, [1] prefers to migrate relatively long-running VMs so that their migration efforts are ensured. Moreover, [21] uses volume-to-size (VSR) ratio of a VM to decide its migration. In this paper, we prefer to migrate long-running VMs. Finally, Algorithm 1 is used to place them onto appropriate hosts that consumes less energy and performance is assured along with aggregation or segregation.

---

**Algorithm 1. VM Placement Algorithm (EPFF)**

**Input:** List of hosts ($H$), List of VM requests ($V$)  
**Output:** Efficient VM placement

1. find runtime efficiency of host $h \in H$ (using Eq. (2));  
2. find the temporal slack of host $h \in H$ (using Eq. (3));  
3. categorise $H$ subject to their runtime efficiencies - $H_{\text{c}}$;  
4. for each $vm \in V$ do  
5. estimate (past) or predict runtime of the $vm$;  
6. match $vm$ to $H_{\text{c}}$ and pick all suitable hosts ($H_{\text{m}}$);  
7. sort $H_{\text{m}}$ in ascending order - temporal slack (Eq. (3));  
8. compute and sort $H_{\text{m}}$ w.r.t ERC values (Eq. (1));  
9. for each $h \in H_{\text{m}}$ do  
10. if $h$ has enough resources and can run the $vm$ then  
11. allocate $vm$ to $h$;  
12. break the loop and pick the next $vm$;  
13. end if  
14. end for  
15. if $vm$ did not fit in any available $h$ then  
16. start new $h$ and allocate $vm$;  
17. else  
18. “$vm$ cannot be allocated”;  
19. “push the $vm$ request into $W$ (waiting queue)”;  
20. end if  
21. end for  
22. return output

---

From implementation perspective, all servers are classified into groups based on their energy consumption and performance (CPU architecture). Workload of particular type is, then, placed on separate server groups, as appropriate. In contrast, workload of any type can be placed on any suitable server in the segregation-based allocation approach. The worst case computational complexity of Algorithm 1 is $O(nm) + T_p$ where $n$, $m$ and $T_p$ denote the number of VMs, hosts and runtime prediction time, respectively. Moreover, $T_p$ is dependent on the workload type, historical data and the prediction algorithm. The best case occurs when each VM is allocated in the first iteration. This also applies to Algorithm
2 with additional time for finding migratable VMs and appropriate target hosts. Also, given that resource properties can change over time; and if a runtime approach is adopted, then, potentially there may be oscillatory or repeatable behaviour, e.g., move VM from host X to Y and then back to X. We can use techniques like CMCR i.e., Consolidation with Migration Cost Recovery [1] or put a constraint to avoid such repeatable migrations. We believe, the proposed VM selection policy (Algorithm 3), that prefers to migrate long-running VMs, ensures to control these repeatable migrations, but not essentially.

Algorithm 2. Consolidation Technique (EPAM)

\[ \alpha_{vm} = C_{host} \cdot d_{vm} - \sum_{k \in N_{host}} w_k (\min \{r_k, r_{vm}\}) - s_{vm}. \]  

\( C_{host} \) denotes the host resource capacity or a notion of VM density [1] which can be computed by the host’s number of cores and VM sizes (without any concerns whether the VM resources are less, more utilised). Further, \( r_{vm} \) and \( s_{vm} \) represent the VM release time (expected runtime) and start time, respectively. Similarly, \( r_k \) and \( w_k \) denote the VM’ runtime and CPU demand on \( k^{th} \) host. In practice, VM release time and start time are determined by users not by administrators. However, we assume that workload type is known, and the VM runtime \( d_{vm} \) is computed as \( r_{vm} - s_{vm} \). \( N_{host} \) denotes the number of VMs on host. Assigning VMs to host with the least \( \alpha \) offers opportunities for switching on/off hosts when it is most cost effective. Hence, \( \alpha \) is measured in CPU \( \times \) time; therefore, it can be easily translated to energy consumption, performance and cost. Note that, \( d_{vm} \) and \( r_{vm} \) aggregate demand w.r.t release time. In addition, we only consider CPU in this formulation, since it is predicated on the assumption that CPU is the largest driver of power consumption [1]. However, there are other components such as GPUs, memory, disks, and network devices whose power consumption is also worth considering. More details about the energy consumption and models of these devices can be found in [22]. After computing \( \alpha \) for each host, all hosts \( H \) are classified into several clusters \( H_{vm} \) using \( \alpha \) and k-Means clustering approach. Next, all hosts in each cluster are sorted in increasing order of their \( \alpha \) values. Lastly, every VM is assigned, based on its runtime (past or predicted), to a suitable host cluster and, subsequently, holding the least \( \alpha \) value.

3.2.2 k-Means Clustering

Using the k-Means clustering approach, we create a set of clusters in order to categorise all VMs. All VMs in a cluster retain comparable features and each VM is mapped to a single cluster. Furthermore, every VM is linked with different types of VM characteristics i.e., runtimes, capacities, workloads (aggregation criteria). To divide these VMs into different categories (\( z \) – selected in advance), we characterise each VM as a point in the multi-dimensional \( R_z \) space – where each point is a VM and coordinates of the point denote VM categories. The inputs of the algorithm, for certain VM characteristics, are the number of categories (clusters) that are specified by its center point. For each characteristic, the cluster centers are points in the \( R_z \) space. In fact, the algorithm selects \( N \) data points to \( z \) clusters. In step 1, centers of the \( z \) clusters are selected randomly. In step 2, the algorithm allocates each data point to the cluster with the nearest center using a particular distance measure such as euclidean distance. In step 3, these cluster centers are recalculated based on the current assignment. The algorithm repeats step 2 and step 3 until no further changes occur [2].

3.3 Modelling Heterogeneity of Infrastructure

In this section, we explain how energy consumption of virtualised hosts and performance of various workloads (and co-located VMs that compete for similar resources) across several heterogeneous hosts can be modelled. These factors are essential to account for as, potentially, they might have impact on users’ costs and service revenues.

[Authorized licensed use limited to: University of Melbourne. Downloaded on November 03, 2023 at 02:42:20 UTC from IEEE Xplore. Restrictions apply.]
3.3.1 Energy Consumption

Energy efficiency of a non-virtualised host could be accurately identified through profiling its various resources for energy measurement. However, the energy consumption of a virtualised host may, possibly, be related to the number of VMs they accommodate. This means that an energy expensive host (virtualised) may, possibly, run a VM more energy efficiently than an energy cheaper, but, non-virtualised host. This relationship could be understood more effectively through relating virtualised and non-virtualised hosts to a bus and a car, respectively. A bus consumes more fuels but still offers cheaper fare than a car. In a similar way, for a particular VM a less energy efficient machine might be more efficient if it can accommodate more VMs. Using the host (non-virtualised) linear power model which is more than 90% accurate, for certain workloads under reasonable assumptions [20], and the most widely used [1], the power/energy consumption of a single VM can be estimated using Eq. (4)

\[
\mathcal{P}_{vm}^h = \frac{P_{idle}^h}{N} + W_{vm}^h \times (P_{busy}^h - P_{idle}^h) \times U_{vm}^h, \tag{4}
\]

where \(N\) is the total number of VMs on a particular host \(h\), \(P_{idle}^h\) and \(P_{busy}^h\) are the power consumed when \(h\) is idle (0% utilised) and fully utilised, respectively. Further, \(W_{vm}^h\) are the host resources (cores) allocated to the VM and \(U_{vm}^h\) is the VM utilisation level. The total energy consumption of the datacenter is the sum of the energy consumed by all hosts; and each host energy consumption could be either: its benchmarked values; a linear relationship to its CPU usage – very similar to Eq. (4); or \(\sum_{vm} \mathcal{P}_{vm}^h\) for all \(m\) number of VMs running on the virtualised host [1]. Since, for the duration of migration there are exactly two VMs running on source and destination hosts which also cost energy. In order to account for migration energy cost, we use the model suggested in [17]. According to [17], energy is largely consumed by transferring the VM memory; and the amount of energy is directly proportional of the VM size (as given by Eq. (5)). We prefer to use this model because it is more than 90% accurate for certain workloads under reasonable assumptions [17].

\[
E_{msg} = 0.512 (VM_{mem}) + 20.165, \tag{5}
\]

where \(VM_{mem}\) denotes size of the VM and parameters of the linear model (\(\alpha = 0.512, \beta = 20.165\)) are computed through empirical evaluation [17]. Besides memory, disk and network states will also consume energy. Moreover, once the duration of a particular VM migration is predicted, then it is also possible to compute the expected energy consumption through multiplying the source (server) and network energy profiles with durations. However, this might not produce accurate estimation compared to the model in Eq. (5) which already accounts for network and disk state costs.

3.3.2 Performance

Various studies suggest that performance of cloud applications or workloads perform quite differently due to: (i) CPU models [4]; and (ii) resource contention [3], [15], [23]. Regarding (i), similar VMs (workloads) run quite strangely even on same CPU models; which may be related to either design (fabrication process), cache levels and/or memory churns. Largely, the distribution of workload runtimes follows a log-normal pattern across different CPU models [4]. Moreover, a particular workload may run quickly on a specific CPU model, but, may run quite sloan on another CPU model. Similarly, a CPU model may run a particular workload quickly, but, another one quite slow. For example, E5430 is faster for bzip2 benchmark than E5507, but, is slower for povray benchmark – as shown in Table 1. Regarding (ii), co-located VMs on a specific host may experience severe performance degradation, particularly, if they compete for same resources (resource interference). The degradation is dependent on the total number of co-located VMs and the workload type they are running on a particular host – as shown in Table 2. In order to model performance variations, we model: (i) CPU heterogeneity as log-normally distributed with respect to workload runtimes; and (ii) resource contention as regression line equation with respect to total number of co-located VMs on a particular host for certain workloads. Moreover, performance of workloads is also affected due to VM migrations; and we account for that, as described in Section 3.1.3. Note that performance, here, refers to sum of all VM runtimes that run workload \(W\) (most suitable to users which translates into costs), and is given by

\[
R = \sum_{vm \in W} Runtime_{vm}, \tag{6}
\]

where \(Runtime_{vm}\) is the wall-clock time (measured in seconds) of each \(vm\) involved in running workload \(W\). Further, users are billed according to runtime of each \(vm\) as described in Section 4.3.6. Similarly, cost of running a particular workload is the sum of all VM costs.

3.3.3 Workloads

Various workloads have different impacts on infrastructure energy consumption, workload performance, and migration durations. Therefore, it is necessary to characterize workload types, even, if real datasets are used or replayed in simulations [1]. An easy way to characterize workloads is to use their resource utilisation levels. For example, CPU intensive workloads would have large impact on CPU utilisation; but not, essentially, on memory or disk usage. Similarly, memory or disk intensive workloads will have little impact on CPU usage. However, in real, scenarios are completely different, probably, due to CPU heterogeneities. Another approach is to use tasks’ priorities, that affects billings, as a proxy to represent workload type [19]. However, this is not reasonable for virtualised workloads [11]. Moreover, our investigation of the Google cluster and Microsoft Azure

<table>
<thead>
<tr>
<th>Workload type</th>
<th>CPU model</th>
<th>Execution times</th>
</tr>
</thead>
<tbody>
<tr>
<td>bzip2</td>
<td>E5430</td>
<td>447 s</td>
</tr>
<tr>
<td></td>
<td>E5507</td>
<td>641 s</td>
</tr>
<tr>
<td>povray</td>
<td>E5430</td>
<td>579 s</td>
</tr>
<tr>
<td></td>
<td>E5507</td>
<td>544 s</td>
</tr>
</tbody>
</table>

TABLE 1

Execution Times (Seconds) of Various Applications Across Different CPU Models [4]
datasets suggests that these workloads (containerised, virtualised) perform quite differently [1]. Therefore, we use monte-carlo simulations to create synthesized workload from real benchmarks workloads that were produced in a real IaaS cloud [4]; and obtain certain features (resource demand and usage, arrival time, submitting users) of the original traces using the laws of statistical distributions and mapping [1]. The synthesized traces follow the features in the original traces.

Table 3 describes the performance (runtimes) of various benchmark workloads (Povray, Namd, Stream) when executed over different CPU platforms [4]. Povray is short-running, Namd is long-running, and Stream is of mixed nature; when run at maximum speed. Note that, stream values originally represent the bandwidth (i.e., data transfer) [4], however, we assume these as durations – since the less data copied, the least time it will take [1]. However, if utilisation levels are normally distributed, then execution times vary. Moreover, distributions of runtimes for a particular workload necessarily follow multi-modal lognormal patterns; where multi-modality relates to CPU architectural heterogeneity. Using laws of lognormal distributions [1], we generated three different synthesized workloads from the reported values i.e., mean (μ), standard deviation (σ), minimum, and maximum, as shown in Table 3. We believe, the generated workloads closely match real workloads; and can be assumed as mix of workloads. Further details on workload modelling and mapping them to real applications’ performance benchmarks are described in [12], [24].

## 4 Performance Evaluation

We assume energy, performance and cost efficient VM placement and consolidation as types of bin-packing problem that can be solved using various heuristics such as first fit, best fit. Energy can be decreased via increasing the resource utilisation levels; that subsequently minimises the number of used servers. Similarly, performance can be ensured either via: (a) relocating workloads to best performing hosts; and/or (b) minimising co-location. In both cases, the proposed scheduler ensures to put similar workloads onto same hosts such that energy and performance efficiencies are achieved. Albeit, techniques like linear programming can be used to come up with an optimal or approximate solution [25]. However, for large-scale systems consisting thousands of servers and variety of workloads, we prefer quickness rather than optimality – as happens in real clouds such as Intel [11].

### 4.1 Experimental Set-Up

In order to evaluate the proposed policies, the CloudSim [7] simulator was extensively modified to simulate a real heterogeneous datacenter as close as possible. For example, classes were added to account for: CPU architectural heterogeneity, performance of co-located VMs, migration costs in terms of energy consumption and performance loss, VM level power consumption, and predicting workload runtimes, migration durations. Details of the extended version of the CloudSim i.e., PerficientCloudSim are elaborated in [24]. Moreover, performance degradation due to migrations, migration durations, and workload runtimes are predicted using various machine learning techniques such as linear regression, SVR and gradient boost trees [11], [16]. Note that, we used the well-known implementations of these algorithms using the sklearn package. The energy consumption of various servers is computed according to SPECpower1 benchmarks. Furthermore, if servers are idle with no workload running (0% utilised), we still assume them as switched on and, therefore, consume their idle power (P_idle). The energy consumption of a single VM and virtualised host is computed using the linear power model which is suggested more than 90% accurate [20].

Our simulated datacenter is a direct modelling of the Google cluster [19] that comprises 12,583 heterogeneous servers that belong to five types, as shown in Table 4. Speeds of various servers were mapped to millions of instructions per second (MIPS) in order to be consistent with the CloudSim. For aggregation-based VM placement, all available servers are grouped into five different clusters – based on these five types of CPU models. For example, all servers of CPU model “E5430” denote a separate cluster. Virtual machines of six various sizes and speeds were assumed running three different kinds of workloads (as shown in Table 3). The utilisation levels of all workloads were modelled as normally distributed with respect to prior studies [20]. Frequencies of VMs, as shown in Table 5, were mentioned in vCPUs (cores), converted to ECUs (EC2 Compute Unit) and mapped to MIPS rating, accordingly. The ECU is described as: “equivalent

---

1. <https://www.spec.org/power_ssj2008/>
TABLE 4
Servers Types and Characteristics for Simulated Datacenter [ECU = CPU Speed (GHz) × Number of Cores]

<table>
<thead>
<tr>
<th>Model</th>
<th>Speed (MHz)</th>
<th>No of Cores</th>
<th>No of ECU</th>
</tr>
</thead>
<tbody>
<tr>
<td>E5430</td>
<td>2,830</td>
<td>8</td>
<td>22.4</td>
</tr>
<tr>
<td>E5507</td>
<td>2,533</td>
<td>8</td>
<td>20</td>
</tr>
<tr>
<td>E5645</td>
<td>2,400</td>
<td>12</td>
<td>28.8</td>
</tr>
<tr>
<td>E5-2650</td>
<td>2,000</td>
<td>16</td>
<td>32</td>
</tr>
<tr>
<td>E5-2651</td>
<td>1,800</td>
<td>12</td>
<td>21.6</td>
</tr>
</tbody>
</table>

CPU capacity of a 1.0-1.2 GHz 2007 Opteron or 2007 Xeon processor" and its rating is per vCPU/core; therefore, the VM total rating is the multiple of cores (number) and ECU rating. The rating is, then, translated to MIPS for consistency with CloudSim as it does not support the notion of ECU. Note that, the large difference in storage capacities of VMs, which ensures heterogeneity, but this will have a clear impact on the migration costs. Performance parameters for servers and VMs (workloads) were taken from real experimental values, as demonstrated in [1], [4]. Various heuristics, that aggregate or segregate workloads using different features such as runtimes, were considered for initial VM placement. At five minutes interval, the optimisation module searches for consolidation opportunities – if utilisation level of a server exceeds 80% or drops below 20% which are two pre-defined threshold values. Our empirical evaluation was accomplished using two different approaches for VM live migration i.e., pre-copy and post-copy. Moreover, workload sizes (runtimes) were transformed to equivalent MIPS over a rating of 2 GHz CPU. From implementational simplification point of view, performance loss or gain was modelled as subtraction or addition of MIPS to the workload size, respectively.

In order to demonstrate the impact of EPC-aware VM placement and optimisation on infrastructure energy consumption, workload performance and service costs, we consider different approaches to VM placement (first fit - FF, energy aware first fit - EFF, energy and performance aware first fit - EPFF) and consolidation with migration (no migration - NO, migrate all - ALL, energy aware migration - EAM, energy and performance aware migration - EPAM) [1]. Note that, VMs selected for migrations are also placed on target servers using these heuristics. In addition, we account for migration energy and performance costs. For example, in ALL approach, all migratable VMs are given chances to migrate; however, in EAM and EPAM those migratable VMs are migrated which can recover their migration costs [1]. Moreover, these policies are implemented using two different methodologies to placement i.e., segregation-based and aggregation-based. The former one ensures that various workloads run in mix (i.e., collocated across a single cluster) while the later one puts similar workloads (based on runtimes, workload types) on same servers (same CPU architectures, similar runtime efficiencies). Similarly, the proposed methodologies have been implemented in two different ways: (i) using past runtimes [1]; and (ii) using certain prediction techniques to predict workloads’ runtimes [16].

4.2 Evaluation Metrics
Data for various metrics, such as energy consumption (KWh), performance or runtime (seconds), total number of migrations, ERC, resource usage statistics, was collected during simulations. Moreover, prediction accuracy is computed in terms of absolute error both for runtimes ($AE_{runtime}$) and migration durations ($AE_{migration}$). The $AE$ denotes the divergence of the estimated value from the actual value in absolute units i.e., seconds and converted to hours.

4.3 Results and Discussion
The results, averaged over ten runs, are shown in Table 6. Our evaluation suggests that workloads run more energy and performance efficiently and, therefore economically, if aggregated onto separate clusters or co-located w.r.t certain metrics and scheduling policies. Moreover, a significant decrease in total number of migrations can be observed; as workloads were initially placed on appropriate servers. Effective allocation techniques are more economical than consolidation approaches; and we suspect, perhaps, this might be a reason that public service providers do not migrate workloads for energy or performance aware computation in their clusters. Furthermore, if migration costs (in terms of energy consumption and performance loss) are considered, then the migrate all approach can be much expensive than the no migration approach. Similarly, if we migrate things only to energy efficient servers, it degrades workload performance and, therefore, may consume more energy due to the existing trade-off between energy consumption and performance (runtime) [1]. These findings are, largely, consistent with previous outcomes [1], [12]. However, if performance is taken into account, significant
energy, performance gains and, therefore, users costs can be saved.

4.3.1 Aggregation versus Segregation

Table 6 shows that aggregation-based placement and/or consolidation (based on workload runtimes) is approximately 9.61% energy and 20.0% performance efficient than segregation-based methodology. The least value for ERC shows the most EPC efficient placement. Fig. 2 describes the percentage improvements, in energy consumption and performance, of using runtime-based aggregation rather than segregation. However, this may not be essentially true for all workloads – as there are certain applications that could perform the best if segregated using other metrics such as workload type, VM sizes etc. For example, if various workloads are placed aggregated (W1 is placed on servers with CPU model E5430, while W2 is placed on servers with CPU model E5-2650, and so on), they result in lower utilisation level of resources, as shown in Table 12. In short, aggregation-based policies offers high levels of datacenter utilisation, with the least performance loss, for particular workloads.

Similarly, if VMs are aggregated on VM sizes, then resources are wasted (stranded resources) [26]. If VM sizes are same, then both approaches are comparable. However, for various sizes of VMs segmentation packs them closely, which: (a) increases resource utilisation (energy efficient); and (ii) higher chances of resource contention (less performance and cost-efficient). Our evaluation suggests that aggregation of VMs, based on workload type, is not ensuring EPC aware placement at all – as shown in Table 12 (observe ERC values for various workloads and methodologies). This is justifiable as similar workloads often compete for same resources which results in worse performance issues. Furthermore, we observed that using past runtimes for aggregation-based placement and migration of workloads always produces

Fig. 2: Percentage improvements in energy and performance using aggregation-based VM placement instead of segregation, using EPFF allocation and EPAM migration [Boosted Gradient Trees prediction].

TABLE 6
Average Results for Various Combinations of VM Allocation and Migration Policies – The Lowest Values are ‘Best’ [± Denotes Standard Deviation, the Least Value for ERC Denotes the Most Effective and EPC Aware Placement Policy]

<table>
<thead>
<tr>
<th>Policy allocation</th>
<th>No. of migration</th>
<th>No. of migrs</th>
<th>Energy (KWh)</th>
<th>Performance (hours)</th>
<th>ERC</th>
<th>Energy (KWh)</th>
<th>Performance (hours)</th>
<th>ERC</th>
<th>Absolute error AE&lt;sub&gt;alloc&lt;/sub&gt;</th>
<th>AE&lt;sub&gt;mig&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Segregation-based placement</td>
<td>Past runtimes</td>
<td>Runtimes prediction</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FF</td>
<td>NO</td>
<td>0</td>
<td>511.93</td>
<td>302.78±0.02</td>
<td>287.2</td>
<td>0</td>
<td>511.93</td>
<td>302.78±0.12</td>
<td>287.2</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>ALL</td>
<td>523</td>
<td>547.23</td>
<td>349.71±0.21</td>
<td>409.6</td>
<td>6390</td>
<td>552.7</td>
<td>356.98±0.29</td>
<td>431.1</td>
<td>0.35</td>
</tr>
<tr>
<td></td>
<td>EAM</td>
<td>321</td>
<td>493.31</td>
<td>271.02±0.09</td>
<td>233.4</td>
<td>4009</td>
<td>526.56</td>
<td>321.03±0.26</td>
<td>331.5</td>
<td>0.42</td>
</tr>
<tr>
<td></td>
<td>EPAM</td>
<td>1021</td>
<td>443.4</td>
<td>211.67±0.1</td>
<td>121.6</td>
<td>1921</td>
<td>461.52</td>
<td>235.76±0.21</td>
<td>157</td>
<td>0.29</td>
</tr>
<tr>
<td>EFF</td>
<td>NO</td>
<td>0</td>
<td>503.39</td>
<td>291.43±0.08</td>
<td>261.7</td>
<td>0</td>
<td>503.39</td>
<td>291.43±0.14</td>
<td>261.7</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>ALL</td>
<td>412</td>
<td>520.04</td>
<td>313.56±0.41</td>
<td>312.9</td>
<td>4522</td>
<td>525.56</td>
<td>320.9±0.51</td>
<td>331.2</td>
<td>0.32</td>
</tr>
<tr>
<td></td>
<td>EAM</td>
<td>2198</td>
<td>511.25</td>
<td>301.87±0.32</td>
<td>285.1</td>
<td>2390</td>
<td>525.97</td>
<td>321.45±0.42</td>
<td>332.6</td>
<td>0.39</td>
</tr>
<tr>
<td></td>
<td>EPAM</td>
<td>1082</td>
<td>444.89</td>
<td>213.66±0.21</td>
<td>124.3</td>
<td>1693</td>
<td>457.28</td>
<td>230.12±0.39</td>
<td>158</td>
<td>0.44</td>
</tr>
<tr>
<td>EP FF</td>
<td>NO</td>
<td>0</td>
<td>465.96</td>
<td>241.6±0.62</td>
<td>166.6</td>
<td>0</td>
<td>465.96</td>
<td>241.6±0.92</td>
<td>166.6</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>ALL</td>
<td>338</td>
<td>443.57</td>
<td>211.9±0.12</td>
<td>121.9</td>
<td>3319</td>
<td>479.24</td>
<td>299.32±0.13</td>
<td>197.2</td>
<td>0.49</td>
</tr>
<tr>
<td></td>
<td>EAM</td>
<td>1502</td>
<td>439.22</td>
<td>206.11±0.44</td>
<td>114.2</td>
<td>1699</td>
<td>460.86</td>
<td>234.89±0.56</td>
<td>155.6</td>
<td>0.5</td>
</tr>
<tr>
<td></td>
<td>EPAM</td>
<td>921</td>
<td>434.77</td>
<td>200.2±0.31</td>
<td>106.6</td>
<td>1256</td>
<td>459.45</td>
<td>233.01±0.34</td>
<td>152.7</td>
<td>0.49</td>
</tr>
<tr>
<td>Aggregation-based placement</td>
<td>Past runtimes</td>
<td>Runtimes prediction</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FF</td>
<td>NO</td>
<td>0</td>
<td>510.76</td>
<td>301.23±0.03</td>
<td>283.6</td>
<td>0</td>
<td>510.76</td>
<td>301.23±0.13</td>
<td>283.6</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>ALL</td>
<td>2898</td>
<td>494.63</td>
<td>279.78±0.73</td>
<td>237</td>
<td>3033</td>
<td>501.41</td>
<td>288.79±0.76</td>
<td>255.9</td>
<td>0.29</td>
</tr>
<tr>
<td></td>
<td>EAM</td>
<td>1677</td>
<td>459.52</td>
<td>233.1±0.56</td>
<td>152.8</td>
<td>1799</td>
<td>485.75</td>
<td>267.98±0.51</td>
<td>213.5</td>
<td>0.38</td>
</tr>
<tr>
<td></td>
<td>EPAM</td>
<td>922</td>
<td>434.45</td>
<td>199.78±0.33</td>
<td>106.1</td>
<td>1209</td>
<td>458</td>
<td>231.08±0.42</td>
<td>149.7</td>
<td>0.4</td>
</tr>
<tr>
<td>EFF</td>
<td>NO</td>
<td>0</td>
<td>495.4</td>
<td>280.81±0.47</td>
<td>239.1</td>
<td>0</td>
<td>495.4</td>
<td>280.81±0.42</td>
<td>239.1</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>ALL</td>
<td>1999</td>
<td>488.51</td>
<td>271.65±0.35</td>
<td>220.6</td>
<td>2777</td>
<td>508.35</td>
<td>298.02±0.33</td>
<td>276.3</td>
<td>0.27</td>
</tr>
<tr>
<td></td>
<td>EAM</td>
<td>911</td>
<td>491.36</td>
<td>275.43±0.21</td>
<td>228.1</td>
<td>1435</td>
<td>503.75</td>
<td>291.9±0.45</td>
<td>262.7</td>
<td>0.5</td>
</tr>
<tr>
<td></td>
<td>EPAM</td>
<td>706</td>
<td>445.16</td>
<td>214.01±0.08</td>
<td>124.8</td>
<td>951</td>
<td>468.38</td>
<td>244.88±0.9</td>
<td>171.9</td>
<td>0.48</td>
</tr>
<tr>
<td>EP FF</td>
<td>NO</td>
<td>0</td>
<td>463.23</td>
<td>238.03±0.11</td>
<td>160.6</td>
<td>0</td>
<td>463.23</td>
<td>238.03±0.18</td>
<td>160.6</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>ALL</td>
<td>2001</td>
<td>462.25</td>
<td>236.73±0.54</td>
<td>158.5</td>
<td>2231</td>
<td>473.65</td>
<td>251.89±0.67</td>
<td>183.9</td>
<td>0.2</td>
</tr>
<tr>
<td></td>
<td>EAM</td>
<td>1109</td>
<td>485.56</td>
<td>267.72±0.39</td>
<td>213</td>
<td>1589</td>
<td>473.21</td>
<td>251.3±0.87</td>
<td>182.9</td>
<td>0.46</td>
</tr>
<tr>
<td></td>
<td>EPAM</td>
<td>799</td>
<td>433.61</td>
<td>198.66±0.64</td>
<td>104.7</td>
<td>988</td>
<td>449.05</td>
<td>219.19±0.55</td>
<td>132</td>
<td>0.33</td>
</tr>
</tbody>
</table>
best results. However, if runtimes and migration durations are being predicted, then inaccurate predictions may lead to worse results even than segregation-based methods. There are no EPC benefits derived from runtime prediction; and using machine learning methods may decrease the metrics of interest. This suggests to further investigate other metrics for aggregation-based resource management in IaaS heterogeneous clouds.

4.3.2 Energy versus Performance Aware Allocation
If we allocate workloads on energy efficient servers (or through energy aware placement policy - EFF), then neither energy nor performance efficiency is assured – since energy efficient servers are not essentially performance efficient. Theoretically, energy efficiency is guaranteed; however, lower performance means longer runtimes and these longer durations translate to more energy consumption (i.e., energy performance trade-off) [1]. Moreover, if workloads are placed initially to energy, performance efficient servers (or through energy, performance aware scheduling - EPFF), then both energy and performance are assured. Fig. 3 shows the percentage improvement, in energy consumption and performance, of using EFF and EPFF allocation polices instead of a simple FF approach.

4.3.3 Energy versus Performance Aware Migrations
Previous research findings, as demonstrated in [1], [12], suggest that migrations are costly and sometimes it might be even more economical not to migrate. Moreover, if a particular workload is being migrated several times, repeatedly, it may suffer from severe performance degradation and, therefore, may consume more energy. Therefore, if migrations are controlled through some methodology e.g., (i) migrate relatively long-running workloads [1]; (ii) migrate to energy efficient servers - EAM; then energy might be saved. Further, if migrations are performed to energy, performance efficient servers (or through energy, performance aware policies - EPAM), then both energy and performance are guaranteed. Fig. 4 shows the percentage degradation or improvement, in energy consumption and workload performance, of using ALL, EAM and EPAM migration polices instead of no migration approach (using boosted trees i.e., B. TREE prediction method). Furthermore, due to the existing trade-off between energy consumption and performance (runtime), migration to energy efficient servers only is not economical.

4.3.4 Impact of Predictions on Energy and Performance
As described earlier, workload runtimes and migration durations play an important role in placement and consolidation decisions, particularly, if their objectives are energy efficiency and/or performance gains. To decide energy efficient migrations, such as CMCR [1] and CPER i.e., Consolidation with migration Energy, Performance Cost Recovery [12], runtimes and migration durations are being compared. Therefore, their predictions and accuracy will have an impact on total number of migrations, which may subsequently affect energy consumption and performance. Fig. 5 shows that good prediction technique (such as boosted trees) may offer relatively accurate results over linear regression, SVR (comparable); and, therefore, almost negligible savings and performance gains. Albeit, the improvement is actually trivial (for certain kinds of workloads), and may even be worse if the overhead of the prediction strategy is taken into account. However, for other workloads’ types, considerably higher improvements were observed. This is, possibly, due to the sizes of different datasets gathered for various applications. These findings are inline with [27] that demonstrates the efficiency of using a simple averaging method over using complex learning approaches. Figs. 2, 3, and 4 indicate that the performance of using prediction techniques to predict runtimes is worse than...
that using past runtimes. For migration scenarios (Fig. 4), the performance of prediction-based approaches in many cases is even worse than no migration approaches. However, the prediction overhead of boosted trees is higher due to its computational complexity. Therefore, if the accuracy difference among the three techniques is small, simple linear regression may be a better choice. Therefore, it is recommended, as a future work, to take the prediction overhead of the three techniques into account to provide more convincing experimental results. This suggests the importance of workload prediction in cost-efficient management of datacenter resources.

4.3.5 Running Containerised Workloads Over VMs
Since, containers are replacing VMs, therefore, it is essential to account for containerised workloads [12]. We are aware that lot of more discussions and experiments is needed to show and discuss the differences between VM-based and container-based cloud infrastructures. Therefore, readers should look for our previous works to understand our observations and findings for containerised workloads [12], [28].

In this section, we describe how aggregation and segregation based placement and consolidation policies would affect energy consumption, performance and costs of workloads that run within: containers directly; or containers that subsequently run within VMs [29]. In addition to earlier experimental set-up, as explained in Section 4.1, we illustrated three container types with characteristics shown in Table 7. We assume that each VM can run several containers. Further, the same allocation policy, which is used to place VMs on servers, was also used to place containers on VMs.

We observed comparable outcomes when containers run on virtualised IaaS resources (inside VMs), as shown in Fig. 6. Albeit, servers were largely more utilised, but, with no benefits. This demonstrates that increased levels of datacenter utilisation may not be always beneficial from energy savings point of view. Moreover, significant performance loss was seen, surprisingly, when containerised workloads that run directly on servers were aggregated based on the workload type. We suspect this might be a possible reason for service providers’ that prefer to segregate their workloads. Unexpectedly, when containers were aggregated onto VMs based on their runtimes; then, besides reduced total number of migrations potential energy savings and comparable performance was achieved. This experiment suggests that, for diverse workload types, segregation-based approaches outperform aggregation-based techniques.

4.3.6 Costs Savings
The total electricity bill, user monetary costs and costs savings (in US dollars - $) are described in Table 8. For this analysis, we assume a PUE2 of 1.10 and energy price of $0.88 per KWH3 that mimic a Google datacenter located in the Oklahoma State, USA. Moreover, we assume that users’ bills are computed at the rate of $0.0017 per second.4 The cost of running a particular user’s workload is $C_{user} = \sum_{run} 0.0017 \cdot \text{Runtime}_{run}$; where the runtime of each VM is in seconds. For certain workloads, service providers could save up to ~21.34% energy costs (bills) using aggregation-based placement techniques instead of segregation. Moreover, users’ monetary costs could be reduced up to ~8.39 to 18.99%.

Although, the least users’ monetary costs would certainly affect the providers’ economics (less profit), however, they can attract more customers which can recoup back these losses (large business). Moreover, the above savings will translate to a million dollars per year for hyper-scale IaaS clouds, such as Amazon AWS and Google, that consist of clusters with more that millions servers to offer resources at large scale. Table 9 shows the percentage of savings possible in energy consumption, performance improvement and users’ costs, when using various techniques in relation to CoLocateMe. It is clear that “CoLocateMe” (aggregation-based policies) offers significant performance improvements and energy savings.

4.3.7 Significance of Results
To demonstrate the, significant, statistical differences between the means of the obtained results using proposed methods and others, we performed the t-test analysis. Table 10 shows

---

**TABLE 7**

<table>
<thead>
<tr>
<th>Container Type</th>
<th>Speed (MHz)</th>
<th>Cores</th>
<th>ECU’s</th>
<th>Memory (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1,000</td>
<td>1</td>
<td>1</td>
<td>128</td>
</tr>
<tr>
<td>B</td>
<td>1,225</td>
<td>1</td>
<td>1.23</td>
<td>256</td>
</tr>
<tr>
<td>C</td>
<td>1,500</td>
<td>1</td>
<td>1.5</td>
<td>512</td>
</tr>
</tbody>
</table>

**TABLE 8**

<table>
<thead>
<tr>
<th>Policy</th>
<th>Energy costs ($)</th>
<th>Users monetary costs ($)</th>
<th>Total costs savings (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Segregation</td>
<td>2202.78</td>
<td>1149.87</td>
<td>-</td>
</tr>
<tr>
<td>Aggregation</td>
<td>1732.65</td>
<td>931.56</td>
<td>18.99</td>
</tr>
</tbody>
</table>
*Allocation Migration*

Values (W = 26 based policies. gains, are significant as compared to other segregation-based policies. The failure of the t-test for FF and EFF policies is, perhaps, due to the overlaps that exist in the collected dataset; however, EFF outperforms FF based on the mean values.

Further, PerficientCloudSim is suggested to produce approximately 98.63% accurate results as compared to a real IaaS cloud [1], [12]. This means that approximately ±1.37% error is expected in our simulated outcomes. Thus, the proposed aggregation-based policy is approximately 9.61±0.13% more energy, and 20.6±0.27% more performance efficient than segregation-based policies.

Table 9 describes that these savings, in energy and performance gains, are significant as compared to other segregation-based policies.

### 4.3.8 Comparative Study

Table 11 offers a comparative study of the proposed CoLocateMe approach to other competing methods including Heifer [3], iAware [18], Granite [30], and IGGA [31]. The results were obtained both for aggregation and segregation methodologies. In case of aggregation, compared with other approaches, CoLocateMe could save approximately 3.02% – 7.22% energy and improve the workload’s performance i.e., 0.34% – 6.6%. The CoLocateMe approach is ~1.11% – 2.54% more energy efficient and 0.58% – 4.48% more performance efficient than other methods when segregation is taken into account. As, CoLocateMe prefers aggregation, therefore, the savings are potentially small in the segregation environment. Furthermore, the proposed approach has lower ERC values than all the closest rivals. The results demonstrate that aggregation is ~11.08% – 24.67% more energy efficient but 0.98% – 8.86% less performance efficient than the segregation. Moreover, aggregation by workloads decreases performance but could be more energy efficient than through aggregating with runtimes. Using various statistical methods, we believe that these outcomes are reasonable against a real IaaS cloud i.e., ~98.99% accurate [24].

### 4.3.9 Generalisation of Outcomes

In order to find consistency in our results and scalability of our proposals, we evaluated the proposed techniques using a variety of heterogeneous dynamic workloads, heterogeneous servers, various metrics for aggregation (such as runtime, workload type), and datacenter sizes. The experiments were carried out using experimental set-up and mathematical models, as described earlier in Section 4.1. In additions, three workload types \(W_1, W_2, \) and \(W_3\) which belong to tasks of three different priorities (0, 2, 9) from Google’s cluster dataset, are also investigated. Furthermore, besides workload runtimes, their type (based on the priority or resource usage – CPU, memory, disk intensive) are considered for aggregation and segregation. We observed that certain workloads, if aggregated using other features such as workload type, may perform ‘best’ using segregation-based placement. However, our findings are largely consistent regarding datacenter and workloads sizes which means that our approach can be scaled for cost-efficient resource management in hyper-scale datacenters such as Google, AWS, and Azure.

Table 12 describes the results which were obtained using previous experimental parameters and set-up, as initially was described in Section 4.1. Largely, we observed that segregation-based VM placement offers fewer opportunities for migrations. Less number of migration opportunities may ensure workload performance, however, it reduces resource utilisation levels. Moreover, workloads aggregated or segregated using various metrics (such as VM sizes, workload type, submitting users) offer variations in energy consumption and performance, therefore, costs. Our evaluation suggests that aggregating VMs based on their workload types in not ensuring EPC aware placement at all. For example, similar VM types may not be tightly packed on servers, in aggregation, and resources are wasted. However, segregation can ensure tight packing of VMs, but, increases resource contention due to co-location. Furthermore, aggregating workloads of similar duration allowing for more servers to be powered down to save energy. Segregation may imply either: (a) having all servers switched on, and minimising the number of VMs per host; or (b) putting the shortest runtime VMs onto hosts with the longest runtime (the greatest runtime efficiency). In respect of (b), workload performance should be better than (a); because there are more servers and resource contention is lessened (the period of the short runtime), rather than a period of time closer to the longest runtime.

We observed that aggregating VMs that have similar types of workloads could lead to high resource contention, interference (and possibly performance degradation that can be ~12.2%) for CPU activity, as shown in Table 12. However, if VM sizes are assumed as running different types of workloads e.g., CPU, memory, disk intensive; then, the contention will be low. Therefore, it is useful to aggregate VM types that have different resource requirements – as this will reduce energy use (~7.51%) and performance overheads (~13.63%), as shown in Table 12. For aggregating and segregating with respect to the workload type, we...
observed opposite findings as compared to VM sizes – segregation is better than aggregation. This type of profiling is particularly relevant in a real-time context; which needs further investigation.

5 RELATED WORK

In cloud computing, resource allocation algorithms use certain features of the infrastructure, and workloads in order to run them in respect of achievable objective (energy, performance, cost). For example, [1] used the host (virtualised) or VM efficiency factor (\(E_f\)) to minimise IaaS energy consumption and improve (at least maintain) the workload performance levels. Other works [20], also use the host energy efficiency metric (i.e., hosts with the least energy consumption) for efficient placement; however, this metric may not accurately measure the efficiency of a heterogeneous virtualised host [1]. However, due to the existing trade-off among energy and runtime (performance), energy cannot be saved with these methods. In such circumstances, performance must be considered during allocation and migration decisions. In the cloud literature, various research, as demonstrated in [11], [15], [18], [30], [32], have considered performance of workloads along with energy efficiency during resource placement and migration decisions. In [30], the authors proposed a scheduling technique, called Granite, to reduce datacenter energy consumption that accounts for CPU temperature. In Granite, a VM is assigned to a server that results with the least increase in workload runtimes prediction [11], [13], [16], but, in practice they are not reasonable – as public cloud workloads differ significantly from private ones. Therefore, an alternative approach is to use their past runtimes. The only reason policy selects VMs from servers based on their temperature over (at least maintain) the workload performance levels. Dabbagh et al. [8] used workload runtimes to place relatively similar-running VMs onto same hosts. In practice, predicting workload runtimes can be a daunting task. Moreover, all workloads of a particular user could be, possibly, placed on same hosts or VMs.

In practice, public clouds are opaque and service providers are not aware of the workloads they are hosting on their infrastructure. Albeit, there are various efforts towards workload runtimes prediction [11], [13], [16], but, in practice they are not reasonable – as public cloud workloads differ significantly from private ones. Therefore, an alternative approach is to use their past runtimes. The only reason which supports this idea is that Google’s tasks that run for...
more than seven hours continue running for several days or even months [19], [26]. This is further evidenced in Microsoft Azure cluster [11]; however, this may not be essentially true. If we assume that clouds are not opaque; then it is possible to predict their runtimes using historical data [13], [14]. For example, Cortez et al. [11] used gradient boosted tree method to predict VMs runtimes in Microsoft Azure cloud. They also found a close relationship among VMs runtimes, submitting users, and job names (logical). Tumanov et al. [13] predicted job runtimes using various characteristics of the workloads in order to automate resource allocation.

A resource level server disaggregation technique, as described in [33], integrates various resources (such as CPU, memory, storage) from multiple servers into a single pool. With server disaggregation it is also possible to run a single VM on multiple servers which provides higher chances for maximising resource utilisation. Moreover, it offers an easy way to enable vertical resource scaling (adding more resources) of VMs. From resource allocation perspective, server disaggregation simplifies the VM scheduling problem to only one dimension. However, aggregation and segregation based VMs placement and consolidation techniques are not explored. Lebre et al. [34] have also discussed various VM placement and consolidation techniques in terms of three different schedulers: centralised, hierarchical and distributed. Tchana et al. [35] suggested software or application migration to achieve energy efficiency in datacenters. Wu et al. [31] also studied VMs consolidation while accounting for energy consumption and migration costs i.e., performance loss in terms of downtime. The authors suggested a consolidation scheme i.e., improved grouping genetic algorithm (IGGA). The swapping criteria ensures that energy costs are reduced through avoiding unnecessary migrations. Jiang et al. [36] proposed an adaptive resource allocation algorithm that dynamically allocates resources to VMs energy efficiently.

In [37], a solution for improved IaaS energy consumption while keeping SLA violations minimum, is proposed. The proposed scheme uses energy aware methods that are founded over adaptive three threshold framework (ATF), policies for VM selection like maximum ratio of CPU utilisation to memory utilisation (MRCU), minimum product of CPU and memory utilisation (MPCU), and maximum energy efficient VM placement (VPME). The results show that optimal energy efficiency and lower SLA violations could be achieved. In [38], the authors offer a survey of proactive methods for VM placement while classifying them as per their application in forecasting procedures. The predicted strategies are efficient and produce minimal VM overheads. An in-depth analysis of predictive VM placement algorithms is illustrated. A dynamic consolidation policy (ETAS) holistically manages IaaS resources through creating a trade-off between computing resources and cooling systems [39]. An improvement in energy utilisation is seen as compared to thermal aware methods.

The peak efficiency-aware scheduling (PEAS) algorithm initially defines metrics like peak power efficiency, and optimal energy usage for heterogeneous servers [40]. Later, it allocates and consolidates VMs over servers so that maximum power efficiency is achieved. It helps to optimise energy usage, overall system performance, and QoS metrics. Similarly, a VM placement algorithm (PLVMP) ensures to achieve better VM performance and balanced server workload between user and service provider [41]. The PLVMP method improves VM placement and creates an effective load balance among various servers. In terms of energy consumption, an in-depth analysis of proactive VM consolidation techniques and algorithm is offered in [42]. In [43], authors focused over maximising the IaaS profits while solving the problem as multi-objective optimisation issue. An evolutionary algorithm is

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Related Work</th>
<th>CoLocate Me</th>
</tr>
</thead>
<tbody>
<tr>
<td>VMs</td>
<td>✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓</td>
<td>✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Platforms</td>
<td>Containers</td>
<td>✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td>Metrics</td>
<td>✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓</td>
<td></td>
</tr>
<tr>
<td>Placement</td>
<td>✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓</td>
<td></td>
</tr>
<tr>
<td>Management</td>
<td>✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓</td>
<td></td>
</tr>
<tr>
<td>Scheduling</td>
<td>✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>[8]</th>
<th>[5]</th>
<th>[33]</th>
<th>[34]</th>
<th>[35]</th>
<th>[18]</th>
<th>[6]</th>
<th>[31]</th>
<th>[29]</th>
<th>[28]</th>
<th>[36]</th>
<th>[1]</th>
<th>[20]</th>
</tr>
</thead>
<tbody>
<tr>
<td>VMs</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Containers</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Energy</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Performance</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Migration cost</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>User costs</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Co-location</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Placement method</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Single</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Distribution</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Hierarchical</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Aggregation criteria</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Runtimes</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Workload type</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>VM size</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Management policy</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Allocation</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Migration</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Sharing resources</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

TABLE 13
Summary of the Related Work, Closest to CoLocateMe, With Respect to Various Evaluation Criteria
suggested that optimises revenue of the IaaS providers
while SLAs are guaranteed.

An overview of state-of-the-art algorithms for energy effi-
cient datacenters and large-scale multimedia services is
offered in [44]. It also outlines key challenges while designing
and managing green datacenters. Further, the authors high-
light several possibilities for green streaming services. In [45],
a continuous replica placement method is proposed that is
based on greedy heuristics. The placement method creates a
new replica while taking into account the optimisation criteria
and various constraints such as energy, costs, etc. Majority
of the above techniques consider segregation-based placement
and consolidation; while aggregation remains relatively unexplored.
Furthermore, with the notable exception of [2], [8],
VM runtimes, sizes and workloads they run, are not evalu-
ated for similar placement and consolidation decisions. The
summary of the comparison between our proposed technique
“CoLocateMe” and other closely related works is given in
Table 13. We believe, Table 13 would help our readers to
quickly identify gaps for further research.

6 Conclusion and Future Work
In this paper, through empirical evaluation we demonstrated
how various approaches to VM placement and consolidation,
and methodologies such as aggregation and segregation,
would affect the energy, performance and cost efficiencies of
large-scale IaaS providers. Our findings show that, for certain
workload types, significant energy could be saved while their
performance is ensured; through aggregating them on same
servers. Moreover, aggregating workloads of similar duration
allows for more servers to be switched off to save energy.
However, if workloads are aggregated based on their types or other
metrics, then they suffer from severe performance degradation.
Our evaluation also suggests that if containers (instead of VMs)
are aggregated based on their workloads types (instead of run-
times), then segregation-based placement methods might
potentially outperform aggregation-based techniques.

Further research is needed to determine what kinds of
workload are not suitable for aggregation, segregation,
and/or migration. Similarly, investigation of workload run-
times, their accurate prediction and other suitable metrics
such as workload type, sizes, is needed for segregation-
based VM placement. Furthermore, there is a need for the
investigation of other metrics-based aggregated and segre-
gated techniques and their potential impact on energy con-
sumption, performance, and costs. In future research, we
will investigate how aggregation and segregation based
resource management would affect oversubscribed resour-
ces. Moreover, the linear power model is hard to take at
face value; as it only covers CPU resource. Modern process-
ors have P-states and C-states (as opposed to merely busy/ idle)
and fairly involved automated switching between them,
which we should consider in future research. More-
over, as a future plan we will work in order to implement
the proposed strategies in a real-world public cloud.

Acknowledgments
The authors are thankful to Changyeon Jo, from SNU,
Korea, for providing us VMs migration dataset.

References
agement in heterogeneous cloud datacenters,” Ph.D. disserta-
energy-efficient VM aggregation and migration framework for
overcommitted clouds,” IEEE Trans. Cloud Comput., vol. 6, no. 4,
[3] F. Xu, F. Liu, and H. Jin, “Heterogeneity and interference-aware vir-
tual machine provisioning for predictable performance in the
[5] Y. Cheng, A. Anwar, and X. Duan, “Analyzing Alibaba’s co-
docated datacenter workloads,” in Proc. IEEE Int. Conf. Big Data,
and J. Wilkes, “Large-scale cluster management at Google with
Buyya, “CloudSim: A toolkit for modeling and simulation of cloud
computing environments and evaluation of resource provisioning
[9] V. Gupta, “Stochastic models and analysis for resource man-
efficient dynamic capacity provisioning in server farms,” Carnegie
and R. Bianchini, “Resource central: Understanding and predicting work-
loads for improved resource management in large cloud platforms,”
Rana, “An energy and performance aware consolidation tech-
Garst, “Jamais plus de regret scheduling with auto-estimated job run-
CMU-PDL-16-104, 2016.
[14] W. Smith, I. Foster, and V. Taylor, “Predicting application run
times with historical information,” J. Parallel Distrib. Comput.,
electricity controls by price feedback,” in Proc. IEEE Int. Conf.
modeling for live migration of virtual machines,” Cluster Comput.,
migration of virtual machines interference-aware in the cloud,”
Format+schema,” Google Inc., Mountain View, CA, USA,
[20] A. Beloglazov and R. Buyya, “Optimal online deterministic algo-
rithms and adaptive heuristics for energy and performance effi-
cient dynamic consolidation of virtual machines in cloud data
centers,” Concurrency Comput.: Pract. Experience, vol. 24, no. 13,
Black-box and gray-box resource management for virtual
[22] M. Dayarathna, Y. Wen, and R. Fan, “Data center energy con-
[23] P. Lettier and J. Cito, “Patterns in the chaos—A study of perform-
ance variation and predictability in public IaaS clouds,” ACM


For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/csdl.