Managing energy, performance and cost in large scale heterogeneous datacenters using migrations

Muhammad Zakarya[1,2] and Lee Gillam[1]

[1]Department of Computer Science, University of Surrey, UK
[2]Abdul Wali Khan University, Mardan, Pakistan
{mohd.zakarya,l.gillam}@surrey.ac.uk

Abstract. Improving datacenter energy efficiency becomes increasingly important due to energy supply problems, fuel costs and global warming. Virtualisation can help to improve datacenter energy efficiency through server consolidation which involves migrations that can be expensive in terms of extra energy consumption and performance loss. This is because, in clouds, Virtual Machines (VMs) of the same instance class running on different hosts may perform quite differently due to resource heterogeneity. As a result of variations in performance, different runtimes will exist for a given workload, with longer runtimes potentially leading to higher energy consumption. For a large datacenter, this would both reduce the overall throughput, and increase overall energy consumption and costs. In this paper, we demonstrate how the performance of workloads across different CPU models leads to variability in energy efficiencies, and therefore costs. We investigate through a number of experiments, using the Google workload traces for 12,583 hosts and 492,309 tasks, the impact of migration decisions on energy efficiency when performance variations of workloads are taken into account. We discuss several findings, including (i) the existence of a trade-off between overall energy consumption and performance (hence cost), (ii) that higher utilization decreases the energy efficiency as it offers fewer chances to CPU management tools for energy savings, and (iii) how our migration approach could save up to 3.66% energy, and could improve VMs performance up to 1.87% compared with no migration. Similarly, compared with migrate all, the proposed migration approach could save up to 2.69% energy, and improve VMs performance up to 1.01%. We discuss these results for different combinations of VM allocation, migration policies and different benchmark workloads[1].

Keywords: Datacenters, Clouds, Energy efficiency, Resource management, Server consolidation, Performance modelling, Algorithms

1 Introduction

Datacenters are the principal electricity consumers in cloud computing, reportedly consuming ∼70 billion kWh in 2014, equivalent to 1.8% of the US total energy consumption, and are projected to account for ∼73 billion kWh by 2020 [1]. It has been suggested that due to techniques like virtualization and consolidation of VMs [2] this figure (∼70 billion kWh) increased by only 4% from 2010 to 2014, which is a significant improvement over the 24% increase from 2005 to 2010. Energy consumption of Information & Communication Technology (ICT) equipment and servers has a negative impact on our environment as they produce Greenhouse gases (GHGs) due to energy usage, which result in increased global warming. Currently, the share of ICT equipment in global GHG emissions is around 1.6%, and it is estimated to be around 2% by 2020 [3]. It has also been reported that for large service providers datacenter energy consumption accounts for more than 12% of monthly operational expenditures (OpEx) and is the fastest rising cost[2] due to increased quantity of equipment, and higher energy prices. For large companies like Google, a 3% reduction in energy cost could translate to over a million

[1] A part of this work is accepted for publication at 13th International Conference on Economics of Grids, Cloud, Systems and Services (GECON-2016), and should appear in Lecture Notes in Computer Science (LNCS), Springer. [this paper used that part as one approach to demonstrate the findings]

[2] http://www.gartner.com/newsroom/id/1442113

dollars in cost savings over a year [4]. National energy supply problems, fuel costs, and global warming all bring the need for green computing into sharper focus. The closure of all nuclear power plants in Germany[3] and France[4] will put pressure on supply in those countries, and the reduction in coal-based power plants in the UK, will result in an expected energy safety margin of just 0.1% in 2017, bringing a very real risk of power outages [5]. If we assume a similar consumption rate to the 1.8% of total energy consumption seen in the US [1], a 6% increase in datacenter efficiency (UK datacenters consume 2 – 3 TWh per year [6]) could represent a doubling of such an energy safety margin in the UK. [1] also suggests that datacenter energy use will remain constant to 2020 due to the move from internal systems to cloud computing. These environmental and economic reasons motivate scholars and industrialists to explore effective methods for saving energy in datacenters, with cloud service providers having the greater need due to having large numbers of such datacenters.

In infrastructure clouds, datacenters comprise large numbers of heterogeneous hosts (virtualized) that cloud customers can use in the amounts they require for as long as they are willing to pay. When a cloud customer makes a request for (part of) a host, a VM is launched on a host selected by the cloud service provider. The user decides how long to run the VM for. This brings two, related, problems: (***a***) due to resource heterogeneity and co-location [7], [8] the performance of VMs varies, and with it, runtimes and costs [9], and (***b***) the unpredictability of users in these on-demand environments can lead to a number of hosts either being idle or running a minimal VM loading, in principle wasting energy as an idle host [i.e. dual-processor 450 W 2U server][5] may consume up to 60% of its peak energy usage [10], [11], although for modern servers this would be less than 55% [12]. In datacenters, (***b***) in terms of idle hosts can be addressed, in part, through approaches such as efficient scheduling and consolidation [2], [13]. However, consolidation involves migrations that can be expensive in terms of both (***a***) and (***b***): In respect to (***a***), if the VM is migrated to a host which performs worse than the source, the increased runtime can decrease datacenter throughput and energy efficiency, and increase agreed (pay per use) user cost. Contradictorily, if the VM is migrated to a host which performs better, then it is also possible that more energy is consumed but for a shorter time, and so costs less. However, in respect to (***b***), there would be additional energy usage due to migration [14]. There is, therefore, a trade-off between energy and performance, and hence cost, which is largely not accounted for in many published models of VM migration. In fact, it can be more costly in terms of energy, performance and hence cost to consolidate (migrate) some VMs.

In this paper, we investigate how migration decisions can be made such that the VM performance improves in a way that the expected performance level is achieved at the agreed cost (pay per use), and energy is saved through consolidation. We extend our previous work that accounts for energy costs involved in the migration, such that VMs are only migrated to more energy efficient hosts in order to recover their migration cost [14]. We explore the impact of various VM allocation policies when combined with different approaches to migration [as described in Sec. 6], on energy efficiency, performance and hence cost. Key to this exploration is to investigate the trade-off amongst performance, energy, and cost. This exploration is conducted through extensive simulations that use the Google workload traces (mapped to three real benchmark workloads) for 12,583 hosts and 492,309 tasks [15] in combination with CloudSim [16]. Following are the novel contributions of the work presented in this article:

1. model resource and workload heterogeneities in the context of a cloud platform [Sec. 4];
2. an approach/metric to balance/measure the trade-off involved in energy consumption and performance (hence cost);
3. an energy-performance-cost (EPC-AWARE) consolidation approach [Sec. 5.1];
4. large-scale simulations using a real dataset from a cloud provider – Google [Sec. 6]; and
5. a review of state-of-the-art energy-performance-cost efficient scheduling techniques in infrastructure clouds [Sec. 2].

The rest of the paper is organized as follows. We offer an overview of the related work in Sec. 2. In Sec. 3, we discuss the VM allocation process as a multi-objective optimization problem

[3] http://www.bbc.co.uk/news/world-europe-13592208

[4] http://www.bbc.co.uk/news/magazine-25674581

[5] http://www.vertatique.com/average-power-use-server

to: (i) reduce the energy usage; (ii) increase or maintain performance level; and (iii) minimize costs through (i) and (ii). We call this approach the Energy-Performance-Cost (EPC) model. In Sec. 4, we briefly explain the focus of this work and establish how to map the Google data to real workload benchmarks from [9], to afford use of the performance parameters for different types of hosts. We elaborate the methodology of our work in Sec. 5. In Sec. 5.1, we extend our previously proposed allocation and migration technique [14], that avoids migrating VMs that would not perform better on the target host. Sec. 5.2 is devoted to experimental methodology. Moreover, Sec. 5.3 describes various performance evaluation metrics. Performance evaluation and the results obtained are explored in Sec. 6. The experimental setup for several plausible assumptions is explained in Sec. 6.1. We validate the proposed migration technique with real benchmarks (mapped to Google workload traces) in Sec. 6.2 and show that our approach can reduce energy consumption, and that the majority of migrated VMs now perform better and continue to save energy and therefore cost. We further discuss and generalize the findings of this study in Sec. 6.3. Finally, Sec. 7 concludes the paper along with several directions for future research.

2 Related Work

Understanding platform (hardware architecture) variation for a specific workload type, initially, is important because the performance of a VM running the workload is primarily determined by the platform, where it is accommodated/baked. The time taken to run a given workload (heterogeneous) will depend on the hardware platform (heterogeneity of the resources and instance classes), and therefore the cost of completing the workload also differs. Xu et al. [17] have discussed several other causes for VMs performance issues such as migration and resource contention when different kinds of workload are taken into account. Furthermore, a systematic review of state-of-the-art is presented that can mitigate VMs performance issues in a: single virtualised host, single datacenter and geo-distributed datacenters. Heterogeneity of computational resources and applications workload is the major cause of such compute performance variation in public clouds. Several studies including [7], [9], [18], on compute performance of EC2 suggests that cloud applications experience significant performance variation, and that can be unpredictable [19]. The authors also observed that similar instances of same instance class on EC2 perform differently for different types of benchmark workload. Generally, there is no 'best-performing CPU model' for all benchmarks workload, neither it is possible to guess performance from CPU age. Typically, in public clouds, compute performance is positively skewed, which indicate the existence of instances with worse performance i.e. higher execution times [7].

The emergence of clouds imposes a significant challenge for applications and service providers. Applications that are running inside VMs are affected by many factors including virtualisation and co-allocation [17]. In the literature [20], [21], [22], [23], different approaches have been suggested to manage datacenter resources in an energy, performance and cost-efficient way. These can be categorized as: (i) resource provisioning; (ii) consolidation with migration; and (iii) methods that describe the trade-off between energy, performance and cost. In the remaining part of this section, we discuss some of these techniques.

(i) Resource Provisioning: Kousiouris et al. [24], investigated the effects of allocation decisions for different kinds of workload and found that the performance overhead can reach up to 150%. The authors suggest that careful selection of the co-allocated VMs and VM placement policies can minimize or even cancel this effect. Furthermore, an experimental study is conducted to investigate the effect of placement decisions, when several VMs are placed on same core/host or neighbouring hosts. In [25], the authors investigated several scheduling policies combined with a consolidation technique to reduce the energy cost which is based on VMs performance level. The authors proposed a performance-based pricing model to increase service revenue and decrease the system energy consumption that can be up to 32%. Hao et al. [26] studied the impact of resource allocation and instance seeking strategies on the system performance and cost, however, energy consumption is not addressed. Furthermore, they suggest

that "resources allocation strategies in the cloud exert a strong influence on the effectiveness of seeking strategies".
The demand on managing the datacenter's TCO (Total Cost of Ownership) is driving researchers to study both energy and performance issues. Lim [27] addressed the cost, energy and performance issues in datacenters and developed a model to estimate the slow-down of applications (workloads) based on the job expected completion time. A similar approach is also suggested in [20] for Hadoop applications. To increase energy efficiency, the authors [27] used a hybrid approach consisting of dynamic provisioning, frequency scaling and Dynamic Power Management (DPM) techniques while meeting the customer SLA's. Although, the proposed hybrid approach could save up to 50% energy as compared to static provisioning with no frequency scaling and DPM, the authors ignored migration and reconfiguration costs. Similarly, the resource and workload heterogeneity is not addressed. Zhang et al. [28] demonstrated that seemingly equivalent platform choices (instance types) for Hadoop cluster in Amazon EC2 results in different application performance (completion time) that leads to various provisioning costs. They considered two types of applications (TeraSort and KMeans) and found that the performance of TeraSort is better on small instances (40 m1.small), while the performance of KMeans is better on large instances (10 mi.large).
Xu et al. [20] suggested that the application performance on A1 VM instances [Microsoft Azure IaaS cloud] can vary by 92.1%, and can even reach up to 280% for the m1 class instances [Amazon EC2] due to resource heterogeneity. Furthermore, the performance interference across VM instances also brings substantial performance variation to applications. To address this issue, the authors presented "Heifer" [20], a hardware heterogeneity and interference aware VM provisioning framework, in order to deliver predictable performance to applications in IaaS clouds. Heifer provisions an appropriate number of VM instances of the good-performing hardware type to applications, and then increases or decreases the number of instances to meet the application performance goals. Heifer achieves an increase of over 16.7% in the provider's revenue, and up to 55.2% decrease in provisioning costs as compared to state-of-the-art techniques. This framework can be extended to use VM resizing and then migrating the resized VM to a good performance host, instead of adding or removing VMs. Moreover, energy consumption is not addressed.

(ii) Consolidation with Migration: Verma et al. proposed pMapper [22], a migration-aware workload placement framework to optimize energy consumption and performance of the application in datacenters. However, it does not consider the cost of turning on and off hosts. The work in [21] suggests "iAware" which migrate VMs, such that both migration and co-location interference can be mitigated holistically, to avoid breaching SLAs. Migration interference occurs when the migrated and other VMs accommodated on source and destination hosts undergo performance degradation. Co-location interference occurs when a migrated VM and other co-located VMs at the destination host suffer from performance losses due to resource contention. The authors suggested that the co-location interference is highly correlated with the number of VMs running on a host. Furthermore, they have proposed two models (demand-supply) to estimate the migration and co-location interference. The proposed approach "iAware" [21] can identify a pair of host (target) and VM with the least performance interference. Experimental results on a real cloud test-bed (for mixed workload) demonstrate that iAware is approximately 16% – 28% more performance efficient than Sandpiper [29] and pMapper – First Fit Decreasing (FFD) [22]. However, migrations are not discussed w.r.t resource heterogeneity and energy consumption. Similar to Cmcr, the migration interference can also be offset if the VM is migrated to a good performance guaranteed host.
In [30], an allocation policy is proposed for consolidated platforms which proportionally scale the provisioned resources according to the workload energy consumption and performance. A feedback mechanism is used to set the performance levels for each application, and if the performance drops a certain threshold, more resources are added to avoid SLA's.
Existing work on VM migration aims to minimize energy consumption without significant impact on workload performance. It is necessary to quantitatively determine the balance between energy savings and workload performance to find the optimal number of VM migrations. Current work has explored the hardware heterogeneity and performance variation of different

instance classes, however, the effect of scheduling techniques and VM migrations is not addressed, when different applications are taken into account – and with the notable exception of [24], [25], [26], this is rarely addressed.

(iii) Trade-off between energy, performance and cost: Jung et al. proposed Mistral [31], a framework that dynamically adjusts VM placement to find a trade-off between application performance (response time), energy consumption, and hosts reconfiguration cost. However, it does not consider the arrival rate of VM requests in its formulation. Production datacenters like Google and Amazon often consist of several generations of hosts with variable capacities, capabilities, and energy consumption characteristics. Meanwhile, the workloads running in these datacenters typically consist of a wide variety of applications with different priorities, performance objectives, and resource requirements [15]. To address this challenge for optimal energy savings and SLA's, Zhang et al. extended their own work in [4] with HARMONY [32]. HARMONY is a heterogeneity-aware framework that dynamically adjusts the number of hosts to strike a balance between energy savings and scheduling delays (SLA's) while considering the host's reconfiguration cost.
Djemame et al. [33] also studied three different VM allocation policies (energy-aware, cost-aware and consolidation) for energy-performance-cost evaluation. The energy-aware policy predicts the VM energy use and places it on a host that will consume less energy. Their results suggest that energy-aware policy consumes 21% less energy than consolidation technique. The findings also demonstrate a trade-off in combination by showing that although energy consumption can be reduced, there is an associated loss in performance. Their experiments do not consider the host's reconfiguration and migration costs. Furthermore, different kinds of workload are not discussed.
Imes et al. [34] demonstrated that different hardware platforms have fundamentally different performance and energy trade-off spaces. As a result, minimizing energy on these platforms requires substantially different resource allocation strategies. Their investigations reveal unexpected differences, that one class of systems requires a *race-to-idle*[6] heuristic to achieve optimal energy consumption, while another requires a *never-idle*[7] heuristic to achieve the same [34], [35]. These consolidation techniques (*race-to-idle*, *never-idle*) have focused on the improvement of resource utilization, particularly CPU utilization and consider little about the performance of workloads. Note that with extensive simulations in this paper, we observed that increase in resource utilization does not always guarantee energy efficiency [Sec. 6.3]. The work in [36] is a notable exception in addressing degradation to workload performance due to the VM co-location and migration. However, it has also ignored how scheduling, resource allocation and migration (consolidation) techniques would affect the energy efficiency, performance and hence cost.

3 Problem Description

In general, the performance of a computing host is not only determined by its processing capabilities measured in terms of core count or clock speed (GHz) but also by all the associated hardware resources such as cache, memory and disk, as well as the network speed. Cloud computing tends to incorporate resource virtualisation in which a host's hardware resources are placed in a resource pool and later shared among multiple VMs through VM sizing [37]. Therefore, in a virtualised environment, the performance[8] of a VM measured in terms of execution time may (or may not) be influenced by the VMs which utilize the same hosts' resources. As an example, when VMs are running similar workloads and competing for similar resources, VM performance can be degraded by up to 67% [7]. It is important to place VMs in the optimal processing host in order to maximize the overall VM performance. However, this would not necessarily be the best approach, since adding more to the same (host) could be bad for it in terms of co-location and performance loss [based on the business paradox][9] as similar workloads

[6] makes all resources available until the task completes and then idles until the next task arrives
[7] attempts to keep the system busy (perhaps not fully utilized) to complete the work just at the deadline
[8] http://www.apmdigest.com/best-practices-to-resolve-resource-contention-in-the-cloud
[9] https://blog.kissmetrics.com/too-many-choices/

will compete for similar resources – referred to as contention. From a service provider's (business) perspective, energy consumption of the processing nodes (hosts) is equally important to the VMs performance due to the existing trade-off between energy and performance [38] and contention will be detrimental to these. Increasing energy and performance efficiency brings at least 3 benefits to service providers: (i) increase profit which allows the providers to invest more money in infrastructure; (ii) decrease CO_2 to get a reputation for datacenters (environmental friendly); and (iii) decrease prices to attract more users that would be to the detriment of profits (i).
Furthermore, in production clouds (e.g., Google and Amazon EC2) the users are charged for the provisioned VM resources and the duration of the service (i.e., job execution time (that does depend on VM performance) → VM runtime). As a consequence of increased VM runtime which may happen due to poor performance of the hosts, the energy consumption of the hosts itself will increase (*even if the host consumes less energy*). Similarly, this might lead to situations where the service provider has to pay penalties for breaching Service Level Agreements (SLAs). These SLAs are more important in applications (with strict deadlines) where the jobs are required to be completed within a short period. Greenberg et al. [39] suggest that performance directly impacts providers' revenue. For example, Google reported approximately 20% revenue loss in an experiment that increased the time to display search results only by 500ms. Amazon also reported a 1% sales decrease for an additional delay of 100ms [39]. Therefore, it is important to make sure that a VM meets its expected performance level (i.e. achieves a service level), and if not, it should be reallocated (i.e. migrated) to a better performing host (at least not reducing the current performance if it is advantageous to do so). The migration decision will be desirable if the newly allocated host is not only a better performing one but also an energy efficient host.
Considering the above diverse opportunities (i.e. increased runtime, decreased performance and increased energy consumption), our work is aimed at developing a model (consolidation) to: (i) predict the energy and performance of a VM; (ii) derive a correlation among the predicted quantities to decide migration; and (iii) finally migrate the VM to achieve better results in terms of the energy consumption and the VM performance. The proposed approach is an attempt to minimize datacenter energy consumption (w.r.t service providers) without reducing the VM performance, even if migrated (w.r.t users). We can express the VM migration as a multi-objective optimization problem which comprises three nominal cost types namely energy consumption cost (Ecc), monetary cost (Mc), and VM performance cost (Vpc). Three entities (i.e., service providers, VMs, and consumers) are involved in the overall process and based on their characteristics each is mapped to a unique objective criterion as described below;

1. Service providers → minimize the amount of energy consumed – Ecc,
2. VMs → achieve their desirable performance level at the agreed costs (to meet SLAs) – in terms of runtime (R), with performance being defined as the inverse of R and the objective here, is to minimize or maintain R – Vpc, and
3. Users → are billed accordingly i.e. minimize cost or maintain the agreed cost – Mc.

It can be intuitively understood that Mc is proportional to R (user is billed according to VM runtime), and therefore, if objective (ii) is satisfied then objective (iii) is also automatically satisfied; hence objective (iii) is ignored in our current work. Consequently, the objectives of our bi-objective optimization problem become to minimize both energy (E) and runtime (R). Mathematically, this can be expressed as Eq. 1:

$$f = \begin{cases} \boldsymbol{min}(E) & where\, E = \sum_{i=1}^{hosts} E_i \\ \\ \boldsymbol{max}(Performance) \Leftarrow \boldsymbol{min}(R) & where\;\; R = \sum_{j=1}^{VMs} Runtime_j \end{cases} \tag{1}$$

The constraints are: (i) each VM is mapped to only one host at a time; and (ii) the number of VMs on a host cannot exceed the host capacity [2].
Gupta et al. [40] used ERP (Energy Response time Product) to capture the trade-off among energy, performance and hence cost, which is widely accepted as a suitable metric to capture similar trade-offs [41]. Minimizing ERP can be seen as maximizing the "performance-per-watt"

– with performance being defined as the inverse of mean response time. In our case, performance is determined through R that can be assumed similar to response time (based on time factor). Hence, we revise the name of this metric to the Energy Runtime Product (ERP). The ERP is given by Eq. 2:

$$ERP = E \times R \tag{2}$$

We assume both E and R are of a comparable magnitude. However, in more complex scenarios, if one dominates the other, then ERP can be expressed as $ERP = \alpha.E \times \beta.R$ (where α and β are the domination factors for E and R respectively) [40], [41]. Theoretically, the objective of the above bi-objective optimization problem is to minimize and evaluate the behaviour of ERP for different scheduling and consolidation approaches with migration techniques, given by Eq. 3:

$$\boldsymbol{min}(ERP) \tag{3}$$

Due to the efforts involved in predicting R, it is impractical to calculate ERP at runtime for each VM that is to be migrated. Therefore, we adopt the above approach to study the behaviour of ERP. Note that, in our previous work [14], the focus of the proposed Fill Up (FillUp) VM allocation policy combined with Cmcr (Consolidation with Migration Cost Recovery) approach was to assign the incoming VM request to a more energy efficient host (based on the host efficiency factor E_f), and continue to allocate requests to the same host unless it is filled. In this case, the approach guaranteed minimum energy consumption and fewer hosts in use, and this creates much greater potential for contention. However, in our current problem, we initially allocate each VM to a particular type of host (according to the information given in Google's data [42]), that might keep more hosts switched on in the datacenter. Hence, the datacenter's total energy use will be high unless we know that a particular type of host is similar or more energy efficient and performs similar or better than another. We investigate through plausible assumptions and event driven simulations in CloudSim [16], how different scheduling and consolidation with migration techniques in a heterogeneous cloud platform, would affect the energy usage, performance, and cost when different kinds of workload are taken into account.

4 Background

In a typical datacenter, sharing a host amongst different kinds of workload would certainly enhance the utilization, thus lowering the economic and environmental impact [27]. Server Virtualisation creates more opportunities for such sharing as it allows multiple VMs of different capacities to share a host resources [43]. However, sharing a host can come at a price, of contention for the available resources that can be higher if workloads are competing for similar resources. Contention could lead to high variation in performance [27], and the problem is widely studied both in theory and in practice [8], [9], [17], [19], [20], [27], [43].
In clouds with heterogeneous resources, which become increasingly likely with scale and longevity of providers, it is common that similar instances will be running on different CPU models with different performance scores (in terms of runtime). Several performance studies like [8], [9], [19], conducted on Amazon Elastic Compute Cloud (EC2), demonstrate that VM runtimes on different CPU models backing a single instance class will have consistent, i.e. largely predictable, performance variations w.r.t the CPU model. The runtimes of an application benchmark within VMs running on a single CPU model shows a lognormal distribution with positive skewness as demonstrated in [8], [9]. However, when running one benchmark across several CPU models it would be possible to imply a distribution but likely not a single lognormal. Several benchmarks across several CPU models would not make much sense to consider a single distribution, as performance across benchmarks does not produce a single ordering on performance. Furthermore, performance variations among similar instance classes in public clouds like Google and Amazon are also observed and compared by several benchmarking (performance) teams, including cloudlook and VPS[10]. There are at least two benefits in understanding such performance variation for different CPU models in a cloud platform:

[10] http://www.vpsbenchmarks.com

1. knowing the relationship between CPU model and runtime means decisions over scheduling can be made to optimize overall energy efficiency;
2. in server consolidation, migration of a VM to a host which leads to an increase in runtime would increase costs for the customer, and as a consequence may violate SLAs.

In respect to (1), if a VM can be migrated to a better performing host, for its workloads, better energy efficiencies may be achieved. But, in respect to (2) if migration would make performance worse, not migrating may be better. Hence, it is important to study the variation in energy efficiencies in heterogeneous clouds. Ideally, a cloud provider should look to the best trade-off between performance, cost and energy requirements; and a user would prefer the best performance (hence cost) for the provisioned amount of resources. However, an appropriate migration approach for resource provisioning and consolidation is needed such that the energy requirements to run the service and the expected performance/price goals can be met.
To evaluate performance variations in large heterogeneous clouds, and their impact on energy efficiency, we need a realistic and representative heterogeneous workload. In the absence of such a workload, we use the Google cluster dataset [15] which contains approximately 25 million tasks over a period of 29 days on a cluster of 12,583 hosts. The cluster machines are heterogeneous and consist of three different platforms (micro architecture and memory technology combinations) and a variety of normalized to maximum compute/memory ratios as shown in Table 1 [44]. A task runs in a Linux container/VM as explained in [44], its CPU requirements are measured in core seconds per second, and the values are normalized with respect to the host with the highest number of cores available in the Google's cluster. Unfortunately, the dataset does not provide exact details of machine specifications. However, we expect that such heterogeneity will translate into variations in energy consumption and performance. We use the VMs runtime as a metric to represent performance variations among different machine types; lower runtimes mean good performance as explained in Sec. 5.3. The dataset still does not have what we need (i.e. different kinds of workload); however each task has a priority and we can use this as a proxy for kind of workload. As the trace providers point out that each task priority affects billing [42], thus we believe that it will accurately reflect the workload type. There are at most 12 different types of priorities that have been grouped by Reiss et al. [44] into three different types of workload: Gratis (free services) (0-1), middle (Batch jobs) (2-8) and Production (monitoring workload) (9-11).

PLATFORM	NUMBER	GCEUs	MEMORY (GB)
A	126	0.25	0.25
B	5	0.5	0.03
	1	0.5	0.06
	52	0.5	0.12
	3863	0.5	0.25
	6732	0.5	0.5
	1001	0.5	0.75
	5	0.5	0.97
C	3	1.0	0.5
	795	1.0	1.0

Table 1: Machines of different architectures in Google's cluster [44]

To simplify concerns initially, we consider only one priority group in each workload type. We can use the other priority workloads later to validate findings. The Gratis, Batch and Production workload includes tasks which have priority 0, 2 and 9 respectively. Each task within a certain workload type can be considered a VM running on one of the aforementioned host types. The machine and platform ID's (in Google data) can be used to get the type of each host on which the resources were provisioned for the VM. We further assume the execution time of each VM as a metric for performance as it is more useful to a user [7]. We selected 492,309 tasks (156,886 tasks from Gratis, 282,464 tasks from Batch and 52,959 tasks from Production) after ignoring those tasks where machine information is missing. Upon visual inspection, the runtime distribution appears to follow essentially a multi-modal lognormal distribution[11], and

[11] A multimodal distribution is a continuous probability distribution with two or more modes (peaks). A single distribution having a mixture of more than one lognormal distributions is called multi-modal lognormal distribution, and the authors in [7], [9] suggest these to model performance variations within the heterogeneous cloud resources.

this may indicate that there are multiple architectures for the same machine class as shown in Fig. 1. It is suggested in [8], [45] that multi-modality relates to CPU architectures and performance is largely determined by CPU model.

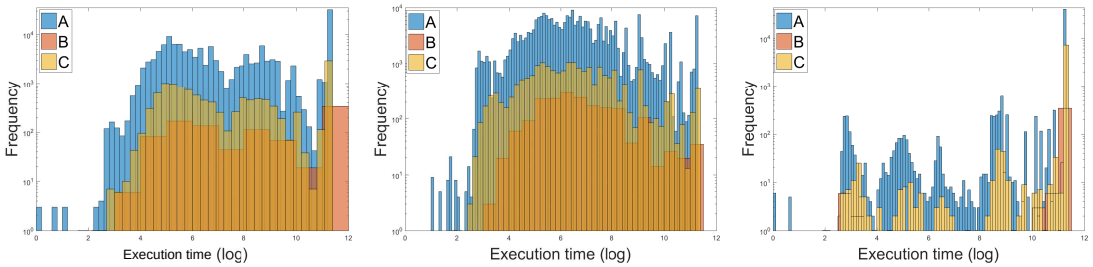

Fig. 1: Workload and execution time (seconds) for machine types A, B & C [from left to right - GRATIS (0), BATCH(2), PRODUCTION(9)]

The performance of all three types of machines is variable and can be fit to at least 2 different architectures for GRATIS and PRODUCTION workloads, based on the number of peaks (local maxima) observed in Fig. 1. However, the runtime distribution of BATCH workload seems to be a uni-modal (single peak) and can be best modelled as a single machine platform. To simplify, each machine type is mapped to $1-4$ different architectures based on the best fit results (likelihood, as explained later) for all three types of workload. By visualizing the distribution, it is easy to identify it as multi-modal. However, there would be several other appropriate ways such as classification and frequency tail (which uses modal value – *mvalue* compared with a threshold)[12] to study distribution modes and decide the number of suggested platform types for each machine class.

We use a multi-modal (lognormal) distribution for each machine class and the goodness of fit (based on likelihood) for such decisions. With more than one peak apparent in the distribution, a Gaussian mixture model is appropriate to estimate the parameters for each multi-modal distribution. The number of suggested architecture types, and fitness parameters, are given in Table 2. The values for mean (μ) and covariance (cov) are given in log because the Gaussian model was used over the log values to represent lognormal distribution. Similarly, the likelihood for each model is calculated through applying Gaussian distribution over the log values of the original data points. We use these parameters (μ and cov) and the number of architectures for each machine class to represent the performance of each machine in our experiments, which are further explained in Sec. 6.1.

Machine Class	Type	Gratis(0) μ	Gratis(0) cov	Gratis(0) likelihood	Batch(2) μ	Batch(2) cov	Batch(2) likelihood	Production(9) μ	Production(9) cov	Production(9) likelihood
A	I	5.429	0.683		6.614	3.395	-5.3e+05	2.853	0.018	
	II	8.581	1.2	-1.68e+05				8.74	0.038	1.243e+05
	III	11.295	0.0001					10.465	0.175	
	IV							11.295	0.0001	
B	I	5.628	0.631		6.757	2.351	-3.52e+03	7.01	12.489	
	II	8.773	0.848	-566.055				11.295	0.0001	1.14e+03
	III	11.295	0.0001							
C	I	5.408	0.483		6.578	3.245	-3.9e+04	3.171	0.085	
	II	8.434	1.133	-1.34e+04				5.388	0.517	2.434e+04
	III	11.295	0.0001					9.425	0.99	
	IV							11.293	0.0001	

Table 2: Parameters for lognormal distributions of hosts performance (suggesting different CPU models) in Google cluster [15]

Fig. 1 (left) shows the distribution of execution time for GRATIS workload on three different machine types (based on local maxima), and each machine can be modelled as though there

[12] http://www.brendangregg.com/FrequencyTrails/modes.html

are three different CPU models. We do not rule out the existence of natural computational variation, however as the data fits to prior findings on performance we can relate these data to heterogeneous infrastructure clouds. These three different architectures perform differently and the performance parameters are explained in Table 2 (Gaussian mixture model is generating the number of distributions). The BATCH workload as shown in Fig. 1 (middle), is a uni-modal lognormal distribution (single peak). However, this can also be represented as multi-modal which is explained in Sec. 4.1. In PRODUCTION workload, machine types A and C can be represented by four different platforms based on the peaks observed visually. However, machine type B is a bi-modal distribution as shown in Fig. 1 (right). Perhaps surprisingly, these ten different machine platforms in terms of different performance behaviour also coincide with the fact that there are ten types of machines in Google cluster dataset [15].

4.1 Relating Google data to Benchmarks

In [7], [8], [9], the authors investigated the performance of different instance types (e.g. m1.small and second generation instances) in AWS EC2 (Elastic Compute Cloud). The authors demonstrated that similar instances running similar or different kinds of workload (BZIP2, POVRAY, NAMD and STREAM) perform differently – dependent on the CPU model hosting the instance. A specific CPU model may perform best for one kind of workload, but worst for another. The results and the parameters to represent performance variation of different CPU models in the previous section were obtained on using several assumptions based on the Google cluster dataset [15]. However, it does not really represent workload dependent performance of hosts in real cloud datacenters. It might be more practical if: (i) we are able to create a mapping between the data we extracted from Google dataset and those benchmark results which are demonstrated in [7], [8], [9], and (ii) we then use the real benchmarks values to represent host performance.

Given knowledge of benchmark runtimes from [7], [8], and [9], we can investigate how to relate and map this with Google data, to determine whether the duration multiples implied from Google cluster data are consistent with such findings. After simulating (monte carlo) the runtimes using the given parameters for different types of workload in [7], [8], and [9], we map it to the Google cluster data on the same scale. Based on similarities between the distributions in terms of the mean (μ) and the standard deviation (σ) of both data (real benchmarks and Google data) as shown in Fig. 2, we are able to create a mapping between the GRATIS (0) workload and the POVRAY benchmark workload running on m1.small instances backed by three different CPU models: E5430, E5-2650 and E5645. The BATCH (2) workload can be mapped to BZIP2 benchmark workload which runs on second generation standard instances (including m3.xlarge and m3.2xlarge) over a single CPU model i.e. E5-2670. If we consider overlapping histograms, then the BATCH (2) workload can be mapped to the NAMD benchmark workload running general purpose m1 class spot instances on 5 different CPU models i.e. E5-2651, E5-2650, E5645, E5430 and E5507. For the PRODUCTION (9) workload, which is mapped to 5 different CPU models, we assume the STREAM workload throughput (data copied in MB/s) as a proxy of instance runtime; and the order of hosts performance is adjusted to MB/s. as lower MB/s means longer runtime to transfer data, therefore the graph in Fig 3 (right) is shown reversed. The PRODUCTION (9) workload is similar to STREAM benchmark workload which runs for longer durations on m1.small instances over four different CPU models: E5430, E5507, E5645 and E5-2650. Different CPU models for each kind of workload which are mapped to the real workload benchmarks (as explained in [7], [8], [9], [18]) are shown in Fig. 3. Note that the CPU models mentioned above are in decreasing order of their performance score i.e. the first one performs better than the second, second one performs better than the third and so on. By mapping these workloads to CPU models, we are able to suggest a preference ordering in terms of performance and performance improvement through migration: a workload landing on an E5-2650 will be best for NAMD, but not as good as E5430 for POVRAY, which would itself be worse for NAMD.

The parameters of runtime distribution for these workloads on these CPU models are extracted from [8], [9] and are explained in Table 3. The distribution of each kind of Google workload is divided to form a multi-modal (lognormal) distribution in such a way (based on the VMs

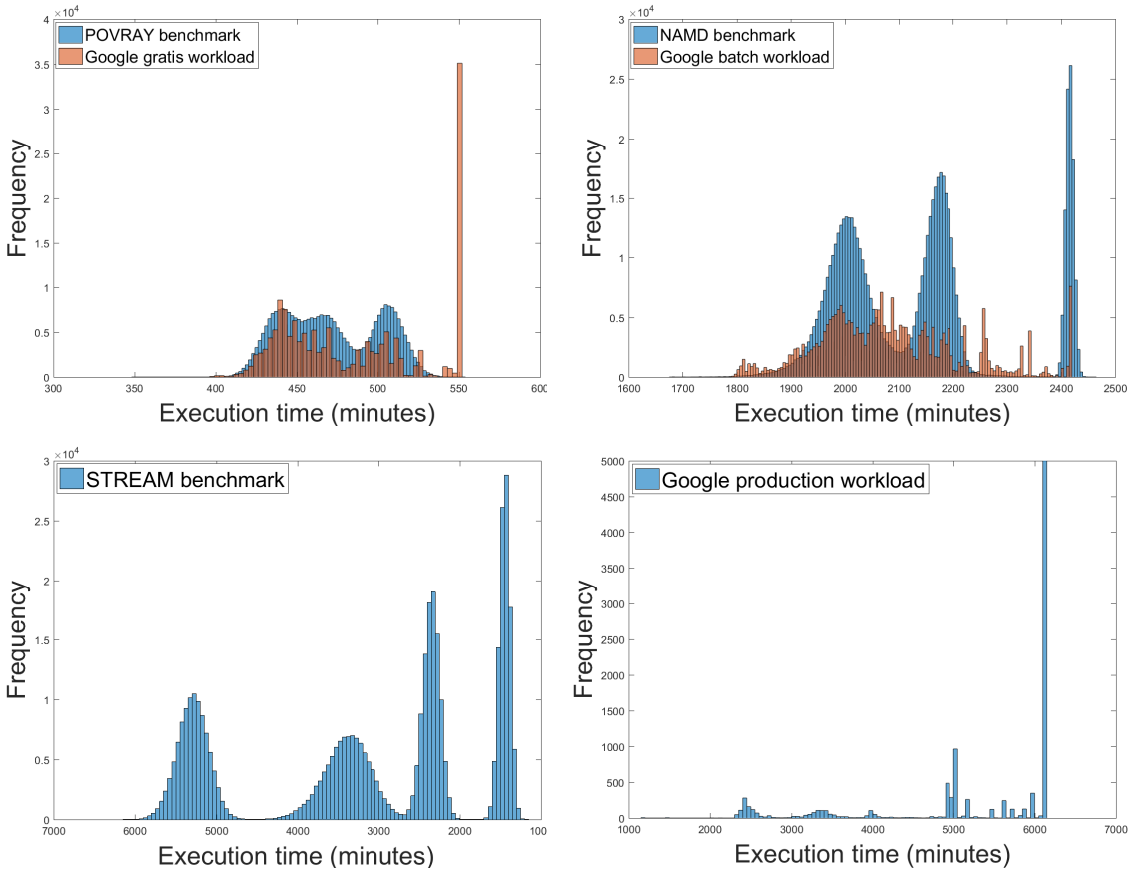

Fig. 2: Mapping of Google Gratis (0), Batch (2) and Production (9) workload to real benchmarks Povray, Bzip2 and Stream [9]

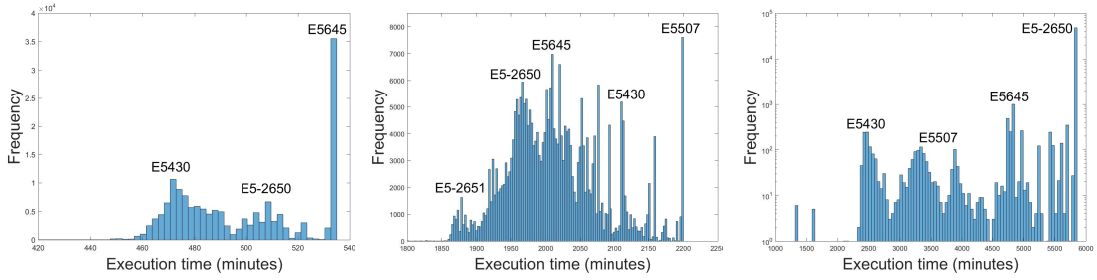

Fig. 3: Google data mapping to real benchmark workload Gratis (0)-Povray, Batch (2)-Namd and Production (9)-Stream (to be interpreted from left to right so E5-2650 is best)

runtime) that it can be closely mapped to the benchmarks. The μ and σ of real benchmark workloads are closely related to the μ and σ of Google cluster data. The Coefficient of Variation (CoV) of each host is calculated with $CoV = \frac{\sigma}{\mu}$, to represent the host performance variations as given in [7], [9].

In Table 3, P is the log value (*as lognormal is equivalent to normal distribution over log values*) of VMs runtime in Google cluster data, that gives an opportunity to distribute the workload among different types of hosts – to represent performance variations. Certainly, there could be other appropriate ways to identify the number of peaks and to divide the runtime distributions (to represent variations in performance) such as the idea of *modal value – mvalue*[13] which is

[13] http://www.brendangregg.com/FrequencyTrails/modes.html

Workload	Benchmark	CPU	Real benchmarks					Google data					
		Model	(μ)	(σ)	Min	Max	CoV	(μ)	(σ)	Min	Max	CoV	P
Gratis	Povray	E5430	439	11	421	467	0.025	438.06	9.42	421	467	0.022	$P < 7.65$
		E5-2650	468	12	451	500	0.026	473.87	11.93	451	500	0.025	$9.75 > P \geq 7.65$
		E5645	507	10	490	535	0.02	498.55	10.44	490	535	0.021	$P \geq 9.75$
Batch	Namd	E5-2651	1994	41.9	1952	2036	0.021	1991	39.51	1800	2040	0.02	$P < 3.8$
		E5-2650	2007	28.5	1978	2036	0.014	1963.4	28.41	1900	2015	0.015	$7.5 > P \geq 3.8$
		E5645	2043	96.4	1946	2140	0.047	1931.4	93.43	1800	2170	0.048	$9 > P \geq 7.5$
		E5430	2160	20.7	2135	2189	0.01	2103.6	22.1	2080	2150	0.011	$10.5 > P \geq 9$
		E5507	2187	18.1	2162	2217	0.008	2191.81	15.69	2150	2200	0.007	$P \geq 10.5$
Production	Stream	E5430	1446	66	1328	1572	0.045	1404.4	44.33	1328	1572	0.032	$P < 5$
		E5507	2348	104	2078	2448	0.044	2346.7	107.21	2078	2448	0.046	$6.3 > P \geq 5$
		E5645	3395	287	2995	4008	0.085	3388.7	238.22	2995	4008	0.07	$11 > P \geq 6.3$
		E5-2650	5294	191	4935	5860	0.036	5294.5	197.52	4935	5860	0.037	$P \geq 11$

Table 3: Different benchmarks runtime parameters for lognormal distribution [8], [9]

compared to a predefined threshold. However, to make it simple, we use visualization and rely on the division points (P), μ and σ as shown in Table 3 for data mapping and integration. Note that the Gaussian mixture model is used to find the number of architectures as shown in Fig. 1. Based on the number of visible peaks, we choose values for P, in such a way that the difference between μ and σ between the datasets (real benchmarks and Google data) is small: the smaller the differences, the more accurate mapping will be.

5 Methodology

In this section, we describe the proposed solution and explain our experimental methodology along with several resource allocation algorithms. Moreover, numerous metrics are discussed to evaluate the performance of various algorithms.

5.1 Proposed Technique

In [14], we proposed an approach to migrate relatively long-running VMs to more energy efficient target hosts. We derived a host (virtualized) efficiency mechanism and a migration cost recovery technique; however, performance variation of VMs due to heterogeneity of resources and workload was not explored. If we migrate a VM to an energy efficient target host, but the VM performance is lower on the target host than the source host, then the energy efficiency effort might be wasted and this could cost the customer more. Hence, in this paper, we extend our previous work and consider the host's performance variations in migration decisions. For two hosts [source (S) and target (T)], where μ and σ are the mean and standard deviation of S; and μ_1 and σ_1 are the mean and standard deviation of T, we can say that an instance backed on T could perform better than S, iff:

$$\mu_1 < \mu \tag{4}$$

In terms of ERP [as explained in Sec. 3], if for a particular VM, the target host ERP (ERP_T) is less than the source host ERP (ERP_S), then both energy efficiency and better performance is guaranteed. Both σ and σ_1 have a key role and would affect the above condition. However, for simplicity, we assume that with given mean values for S and T, we can differentiate between their performance levels. Theoretically, migration of some workload (VM) from S to T [if S and T are of different platforms] can be modelled with z-score normalization (normal distribution), and then converted back to an equivalent lognormal distribution. The z-score (standard score) as given by Eq. 5, can be used to calculate the probability of a score (x) occurring within a normal distribution. Furthermore, it also provides a way to compare two scores that are from different normal distributions.

$$z = \frac{x - \mu}{\sigma} \tag{5}$$

For lognormal distribution, x must be replaced with $log(x)$ according to the definitions of normal and lognormal distributions. The following Eq. 6 can be used to find runtime of the migrated VM (estimated) on T with given μ and σ [lognormal distribution].

$$\frac{log(x) - \mu}{\sigma} = \frac{log(x_1) - \mu_1}{\sigma_1} \tag{6}$$

where x and x_1 are the expected runtimes of the migrated VM on S and T respectively. The left and right sides of Eq. 6 relate to the z-scores of source and target hosts, respectively. This formulation allows us to calculate the probability of a score (expected increase or decrease in runtime of the VM on target host) occurring within a lognormal distribution of the performance variations due to resource heterogeneity. The above Eq. 6 can be rewritten to find the expected runtime of the migrated VM on T, as Eq. 7:

$$x_1 = exp\left(\sigma_1 \times \left\{\frac{log(x) - \mu}{\sigma}\right\} + \mu_1\right) \tag{7}$$

If $x_1 < x$, the VM would perform better on T, and migration can proceed. Otherwise, the migration would not be cost effective and the decision is not to migrate. We combine this with the host efficiency factor proposed in our previous paper [14] into the migration decision. So, if T is more energy efficient and the performance of the migrated VM is better than on S, then there are more chances for the VM to recover its migration cost and be more energy-performance-cost (EPC) efficient. The steps are described in Alg. 1. Moreover, if there are several VMs suitable for the migration, then the VM, which runs for longer duration (previous runtime), is migrated first. The list of all migratable VMs is sorted in decreasing order of their previous runtimes [as described in Alg. 2].

Algorithm 1: ENERGY-PERFORMANCE-COST (EPC-AWARE) MIGRATION

Input: vm (that is to be migrated), H_{source} and H_{target}
Output: return migration decision d

Note: In our implementation, we use R_{past}, instead of $R_{remaining}$, which is already known – [as discussed in [14] ($R_{remainimg} = R_{total} - R_{past}$)];
$d \leftarrow$ FALSE;
Estimate the remaining runtime $R_{remaining}$ of vm [*assuming it is possible*];
$ERP_{source} = E^{vm}_{H_{source}} \times R_{remaining}$
[$E^{vm}_{H_{source}}$ is the energy consumed by vm on host H_{source}];
$ERP_{target} = E^{vm}_{H_{target}} \times R_{remaining}$
[$E^{vm}_{H_{source}}$ and $E^{vm}_{H_{target}}$ are calculated using the power model presented in [14]];
if $ERP_{target} < ERP_{target}$ **then**
    $d \leftarrow$ TRUE [*i.e. migrate the VM using the migration model in [14]*]
    [*the migration model in [14] accounts for migration costs in terms of energy consumption, migration duration and performance degradation*];
end if
return d

Empirical evidence has shown [27], [34], that configuring and executing workload across multiple resources (i.e. segregation and/or parallelism) provides greater energy efficiency than working with only a single resource. However, as shown in [34], scheduling heterogeneous applications on heterogeneous hosts would provide different energy efficiency and performance trade-offs, so each will need a different scheduling strategy to minimize energy while meeting performance objectives. The migration process also introduces performance degradation that can be up to 10% of CPU utilization (of the host overall), according to the empirical evaluation of real benchmarks demonstrated in [46], [47]. The downtime and performance degradation is dependent on the application behaviour and resource utilization that can be approximated by the dirtying rate of the VM memory pages [14] in shared-disk based systems. The average performance degradation as given above (∼10%), which also includes the downtime, is estimated for web applications with dynamic workloads. The performance degradation experienced by a VM (vm) is estimated with Eq. 8:

$$PERF_{degrade_{vm}} = 0.1 \times \int_{t_0}^{t_0+T_{m_{vm}}} u_{vm}(t) \quad dt \tag{8}$$

where $PERF_{degrade_{vm}}$ is the total performance degradation by vm, t_0 is the time when the migration starts, $T_{m_{vm}}$ is the time taken to complete the migration that can be easily estimated using Eq. 9; and $u_{vm}(t)$ is the CPU utilization by vm.

$$T_{m_{vm}} = \frac{vm_{memory}}{vm_{bandwidth}} \tag{9}$$

Note that degradation of performance increases the VM (workload) execution time, as explained later in Sec. 6.1. Several other papers like [21], [48], [49] have demonstrated that the migration of a VM has a negative impact (workload dependant) on the performance of the migrated VM, on both source and target hosts, which in turn affects the performance of all hosted VMs[14]. This impact could be worse if a single host is experiencing the migration of several VMs at the same time. Therefore, well-known resource management tools like Hyper-V[15] do not support migrating more than one VM from a single host at the same time. In this paper, we only migrate VMs from underutilized hosts. Therefore, it is reasonable to assume that except the migrated VM itself, these migrations will not affect the performance of other co-located VMs.

5.2 Experimental Methodology

The consolidation (migration) process is considered an optimization problem with the objective to minimize the number of hosts in use. Every 5 minute, the optimization is performed based on the current utilization level of all hosts, in three steps; (i) *VMs selection:* Every host is observed and if its current utilization is less than a predefined lower threshold value ($Threshold_{low}$), for example 20%, all VMs accommodated on this host are selected for migration. If there is more than one VM for migration from this host, then the proposed VM selection algorithm [Alg. 2] gives priority to the one which is running for longer (migrate one VM at a time from a single host to minimize performance loss). Alg. 2 can be assumed as the core optimization module in resource consolidation [46]. (ii) *hosts selection:* The migration policy selects a most suitable host from a list of all available hosts that can accommodate these VMs. However, to minimize the number of hosts in use, it excludes: (a) those hosts which are not in use (switched off); and (b) hosts which are intended to go into the idle/switched off state (running but with no work on them). (iii) *placement:* The list of selected VMs is arranged in increasing order of their past runtime (R_{past}) to migrate long-running VM first. Finally, a VM allocation policy is used to reallocate all VMs, because VM placement is a sub problem of the migration process [46]. However, it could be of interest to service providers to know whether a single heuristic for both placement decisions is more efficient or if two different heuristics are more economical and energy efficient. In Sec. 6.3, we discuss more about policies for initial allocation and migration placement. The following scheduling heuristics are examined in this paper, both for initial VM allocation and migration placement.

(i) Round Robin (RR): The RR policy places each VM on the next available host based on the concept of a circular queue. In a circular queue (linked list), the last node is connected back to the first node to make a circle. RR records the last position ($host_{lastvisited}$) the scheduler visited and places a new incoming VM on the subsequent host. Therefore, RR seeks to use the fullest extent of available hosts, irrespective of utilization. If more hosts are underutilized, it will create more migration opportunities during consolidation, and increasing the number of possible migrations could lead to higher energy consumption.

(ii) Random (R): A typical R approach picks a random host in the datacenter and checks if the host can accommodate the VM or not (suitable). If not, then it picks another one and the process continues until a suitable host is selected. An improved variant of R in [52] finds a list of all suitable hosts first, and then selects a random host from the list to place the VM. To compare the output of different allocation and migration techniques, it is important that

[14] https://blog.zhaw.ch/icclab/an-analysis-of-the-performance-of-live-migration-in-openstack/

[15] http://kevingreeneitblog.blogspot.co.uk/2010/12/automatically-live-migrate-multiple.html

Algorithm 2: VM SELECTION ALGORITHM

```
Input: hosts list (hostsList)
Output: migration list (migrationList)
for each host ∈ hostsList do
    vmList ← host.getVmList() [vmList is a list of all migratable VMs];
    vmList.sortDecreasingPreviousRuntime() [sort vmList in decreasing order of their
     previous rutimes];
    hostUtil ← host.getUtilization();
    while hostUtil < Threshold_low do
        for vm ∈ vmList do
            migrationList.add(vm);
        end for
    end while
end for
return migrationList
```

during each experiment, R allocates similar hosts (class) to run VMs that can be implemented using *seed*.

(iii) Best Resource Selection (BRS): The BRS algorithm [50] finds the host(s) with the least available slots (more utilized) and allocates them first. BRS technique ensures minimum number of migrations, so should save energy if low utilized hosts could be switched off later. BRS places a VM on a host with the least free capacity which also maximizes resource utilization.

(iv) Minimum Power Difference (MPD): In MPD [51], every VM is allocated to the host which will consume the least energy to run the VM. MPD is based on the concept of Best Fit Decreasing (BFD) heuristic and is used as the primary model of energy savings in the well-known cloud simulator CloudSim [53]. MPD is a modified BFD off-line heuristic[16] that sorts all VMs in decreasing order of CPU utilization and allocates each VM to a host that increases its energy usage the least, selecting the most energy efficient host first, based on the linear power model. However, this off-line heuristic is suitable only for those VM requests whose resource usage are known in advance. LAGO et al. [50] demonstrated different VM allocation policies including RR, BRS and LAGO and found MPD one of the more competent energy efficient techniques in heterogeneous clouds.

(v) First Fit (FF): This heuristic considers the available hosts in a sequential order for placement i.e. for n number of hosts the *getOrder()* procedure always returns the sequence $\{0, 1, 2, ..., n\}$. Effectively, this tries to place the VM onto the first host in the list that has enough space to accommodate it. First, this ensures that a lower number of hosts is utilized, which minimizes migrations and saves energy. Secondly, more energy can be saved through consolidation if idle hosts can be switched off [14].

(vi) LAGO (LAG): LAGO allocator [50] combines the BRS, MPD algorithms and uses the power after allocation (*paa*) technique to select a more energy efficient host for the VM. If there are several energy efficient hosts, then the host with the highest CPU utilization is selected. Host with the lowest utilization is not selected because of the possibility that it can be switched off later to save more energy. The authors evaluated the hypothetical results of this approach and compared it to several other heuristics like RR, BRS and MPD. Their results shows improvement in almost all scenarios compared to the MPD and other algorithms. However, for small homogeneous datacenters there is not a clear significant improvement. For large heterogeneous datacenters, this improvement can reach up to 6.1%, which suggests that the larger the datacenter, the larger the difference between LAGO and MPD.

[16] although it can be considered as an on-line heuristic at 1 second interval

(vii) CARLO (CAR): Mastroianni et al. [52] proposed ecoCloud, where each host takes a decision either to accept or reject an incoming VM placement request, based on the Bernoulli trial against its available resources (CPU and memory). The datacenter manager broadcasts[17] the VM placement request to all available hosts, and then each host responds to the datacenter manager if it can accommodate the VM (*Bernoulli trail is successful*) [54]. The datacenter manager then places the VM on one of the hosts (those which responded positively) randomly. The authors demonstrated that their proposed algorithm performs better that BFD (hence MPD), especially when the resource demand is high.

(viii) Fill Up (FU): The FILLUP approach finds the host(s) with the least available slots (more utilized) first, and then uses the host efficiency factor (E_f) as proposed in [14], to ensure that more energy efficient hosts get used first. The on-line behaviour of FILLUP heuristic differentiates it from other approaches like LAGO, BRS and MPD. In [14], we evaluated the efficiency of FILLUP approach in terms of least energy consumption and number of migrations.

Algorithm 3: INSTANCE TYPE SELECTION ALGORITHM

Input: Instance request (R), available instance types (INSTANCES)
Output: Return a suitable and cheap instance ($Instance_{suitable}$)

```
1  Instance_suitable ← INSTANCES_maxCores [assume that the larger instance is suitable];
2  price ← Price.INSTANCES_maxCores [on-demand price per hour];
3  minPrice ← Price.INSTANCES_minCores [on-demand price per hour];
4  for VM in INSTANCES do
5      if VM can finish R within time (is suitable) then
6          if VM_price < price then
7              Instance_suitable ← VM;
8              price ← Price.Instance_suitable [on-demand price per hour];
9          end if
10     end if
11     if Price.Instance_suitable = minPrice then
12         break for loop;
13     end if
14 end for
15 return Instance_suitable
```

Initially, every VM request goes through Alg. 3, and a suitable instance type is selected [steps 1 – 5]. A suitable instance is defined as the one that could execute a particular workload within the deadline. If there is more than one suitable instance available, then a cheaper one is selected from the list (price per hour) [steps 6 – 10]. We assume that an instance having the highest number of cores ($\text{INSTANCES}_{maxCores}$) is always suitable to run a particular workload. Moreover, an instance having the fewest number of cores ($\text{INSTANCES}_{minCores}$) is, comparably, cheaper than the other instances. We then search for other suitable instances whose prices are less than $Price.\text{INSTANCES}_{maxCores}$ (the price of instance having the highest number of cores i.e. $\text{INSTANCES}_{maxCores}$). In order to increase the efficiency of Alg. 3, when an instance with the lowest cost is observed, it is selected immediately and the search process is terminated [steps 11 – 15]. Note that the runtime of Alg. 3 is dependent on the number of instance types offered by the cloud providers – which are fixed and limited (possibly few) e.g. Amazon AWS instance and Google machine types. The instance is then placed on the appropriate host type (per VM request) by one of the allocation policies. Although this work does not focus on instance cost or propose any price model, we do use simple prices to demonstrate the effects of performance variation. We use Amazon's prices for each VM type based on the reserved prices i.e. for one year with no upfront in US East region (N. Virginia)[14] as of September 2016; given in dollars

[17] in large datacenter, servers may be distributed among several groups of hosts, and the request may be broadcast only to one group, which minimizes the broadcast overhead

per hour – as shown in Table 5.

In order to implement our proposed technique in Sec. 5.1 and the above policies, we made several changes to CloudSim. To represent performance variation (runtime) of host(s), we extended the abstract migration class to calculate the increase or decrease in VMs runtime after migration. An extended class of VM scheduler considers the performance degradation due to migration ($\sim 10\%$) both on source and target hosts. Similarly, each VM allocation policy extends the abstract VM and VM allocation policy class as described in [14]. Additionally, we extend the host class to account for: (i) the host reconfiguration (switching on/off) costs (energy consumption) and (ii) associate each host with a type (platform) so that each VM is initially accommodated on a known host type (when reading VM request from Google dataset). To compare the efficiencies of various VM allocation and migration techniques, in terms of energy consumption and performance, we use various evaluation metrics as described in Sec. 5.3.

5.3 Metrics

The metrics of interest are: the number of migrations, average number of hosts used to run the VMs, total datacenter energy consumed and the average VM execution time R_{avg} (performance). An overall calculation of datacenter efficiency, D [14] measures the efficiency of a scheduling approach at datacenter level (overall). This accounts for the occupancy rate (% slots filled i.e. VM density), the number of hosts 'switched on', the number of 'idle' hosts (idle energy consumption), and a factor of energy efficiency E_f in respect to the efficiency of hosts in use.

$$VM_{density} = \frac{VMs_{onHost}}{Host_{capacity}} \tag{10}$$

$$D = \left(\frac{\sum_{hosts} VM_{density} * E_f}{Hosts_{used}} + \frac{\sum_{hosts} Hosts_{unUsed} * E_f}{Hosts_{unUsed}} \right) \tag{11}$$

The first part of Eq. 11 determines how many efficient/inefficient hosts are utilized and to what levels they are used. The second part represents how many efficient/inefficient hosts are idle and consume their idle power. The host efficiency model presented in [14], is used to calculate E_f for each host, which represents its energy efficiency. Lower values for D represent a more efficient datacenter with VMs running on a minimum number of most energy efficient hosts, and hence also offers potential for hosts to be powered off. Unfortunately, D is not relevant to the system performance, and we need other metrics to measure performance.
We assume each workload type as a single job and use its execution time (wall clock time - R) as a proxy to denote the system performance score. And we use R_{avg} to compare different allocation and migration policies w.r.t performance. Better performance implies being able to execute the given workload more quickly and thus reduces costs for customers. A popular metric for system energy efficiency against performance is performance per watt (PPW) which is used by green500[18] list to rank supercomputers. *"PPW measures the rate of computation that can be delivered by a computer for every watt of power consumed"*. PPW is given by Eq. 12.

$$PPW = \frac{MIPS/GigaFlop}{Watt} \tag{12}$$

where Flop stands for floating-point operations per second which is a standard measure of computing power. A higher value for PPW represents better energy efficiency in performance, and hence lower cost. A similar metric, performance to power ratio (PPR), is used in SPECpower[19] benchmarks to measure the system energy efficiency against performance, based on the system throughput (SSJ_OPS). SSJ_OPS is defined as total number of workload operations (in Flops or MIPS) performed in one second, given by the following Eq. 13.

$$SSJ\_OPS = \frac{Workload_{operations}}{Runtime_{total}} \tag{13}$$

[18] https://www.top500.org/green500/
[19] https://www.spec.org/power_ssj2008/

PPR is given by the following Eq. 14.

$$PPR = \frac{SSJ\_OPS}{Power_{consumed}} \tag{14}$$

In this paper, we use PPR to measure the energy efficiency against performance and hence cost. Note that there are other performance metrics that also demonstrate the existing trade-off between energy consumption and performance (hence cost). For example, Gupta [40] proposed Energy, Response time Product (ERP), also known as the Energy Delay Product (EDP): (i) to capture the trade-off between performance (response time) and energy, and (ii) to compare different consolidation policies. However, ERP does not take the VM runtime, which represents performance (hence cost), into account.
Furthermore, to assert what is best, we need a combined (single) value or metric for the above bi-objective measure of energy consumption and performance. Therefore, we use H as a single measure of both energy consumption and performance. First, we put both numbers on the same scale against the maximum (normalized using min-max technique) [Eq. 15]. Then, the harmonic mean (H) of both normalised values produces a single value allowing for the best to be determined [Eq. 16].

$$E_{norm} = \frac{E - E_{min}}{E_{max} - E_{min}} \quad and \quad R_{norm} = \frac{R - R_{min}}{R_{max} - R_{min}} \tag{15}$$

$$H = \frac{2 \times E_{norm} \times R_{norm}}{E_{norm} + R_{norm}} \tag{16}$$

where E and R represents energy consumption and performance (runtime), respectively. Lower H is preferable, as a combination of low energy use and low runtime (high performance). With that, interpreting results is straightforward: the smaller the H, the better the approach. It is also readily possible to weight one number's importance over another (preference) and then use weighted harmonic mean instead of H, but we do not address this in this paper.

6 Performance Evaluation

As explained in [14], VM allocation (in server consolidation) is a type of bin-packing problem which can be solved using various heuristics that may not ensure optimal results but are fast enough to deal with large problems [2]. It is possible to consider an analogous VM packing problem as moving from a given datacenter state to an ideal state, which should be one using the fewest hosts. We achieve a datacenter state by implementing scheduling heuristics such as Round Robin (RR), Random (R), Best Resource Selection (Brs) [50], Minimum Power Difference (Mpd) [51], Lago [50], Carlo [52], First Fit (FF) [46] and Fill Up (FillUp) [14], with VM packing then needing to guarantee energy through this, cost efficiency is assured (as explained in Sec. 5.1) and the performance is improved or maintained. To evaluate the effect of scheduling policies on VM performance, energy and user cost, we consider the following migration strategies: (i) no migration - No; (ii) dynamic consolidation (all possible migrations) - All; (iii) migrate for better performance - Perf; (iv) Cmcr (runtime-based consolidation technique that migrates relatively long-running VMs to better demonstrate the trade-off between runtime (hence cost) and energy [14]); and (v) migrate for both better performance and Cmcr - Perf+Cmcr.

6.1 Experimental Set-up

A cluster (simulated) of 12,583 heterogeneous hosts, which consists of different types of architecture (varying performance) and hardware specifications – as shown in Table 4 - is available to execute three different types of benchmark workload. These hosts are subdivided by architecture based on the workload type they execute as mentioned in Table 3. The hosts (simulated) are configured based on assumptions that Amazon had certain kinds of commonly available

machines in their infrastructure cloud (EC2), when the experiments in [7], [8] and [9] were performed. The hardware specification and energy consumption values for these hosts are taken from SPECpower[20] benchmarks. The hosts are created based on decreasing order of their performance. For example, if there are three hosts (with different platforms) of a single type, then the first one performs better than the second; second performs better than the third; and so on. We are aware that such an order would affect the results obtained here; however, this is not within the scope of this paper. Our simulation consists of six types from Amazon's instance classes as shown in Table 5. These are ranked (in terms of performance and resource requirements) according to Amazon's description of their VM performance rating – ECU (EC2 Compute Unit), which is described as: "equivalent CPU capacity of a 1.0 – 1.2 GHz 2007 Opteron or 2007 Xeon processor"; its performance variation is about 20% (1.0 – 1.2 GHz) [26]. The ECU rating is per core, so the total rating is given by the number of cores multiplied by ECU rating [8]. These are arranged in increasing order of memory size (as shown in Table 5), which makes it possible to provision enough resources requested by the user with instance selection Alg. 3. Such a setup decreases the user cost and increases the availability of resources for sudden increases in demand.

CPU MODEL	SPEED (MHz)	NO OF CORES	NO OF ECUs	MEMORY (GB)	P_{IDLE} (Wh)	P_{MAX} (Wh)	TOTAL AMOUNT
E5430	2830	8	22.4	16	166	265	
E5507	2533	8	20	8	67	218	
E5645	2400	12	28.8	16	63.1	200	12583
E5-2650	2000	16	32	24	52.9	215	
E5-2651	1800	12	21.6	32	57.5	178	
E5-2670	2600	16	41.6	24	54.1	243	

Table 4: Host characteristics for Amazon's cloud [7], [9] [idle (P_{IDLE}) and maximum power consumption (P_{MAX}) of hosts are taken from SPECpower benchmarks][15]

To address a cloud context, each task is assigned a single, notional, VM that maps to Amazon's instance types. We assume that hosts are comparable by a single measure which allows for performance ranking, for which we adopt CloudSim's use of Million of Instructions Per Second (MIPS) as a proxy; we would not endorse this as a good performance indicator for real systems for a number of CPU architecture and workload comparability reasons. One approach to VM sizing is to assign a VM as a single core for the maximum value 1, half a core (hyperthread) for 0.5, and assume that higher VM gearing leads to a quarter of a core for 0.25. But to address allocation more flexibly, along lines of certain cloud providers, we map CPU frequency for the hosts given to Amazon's ECUs as: 1 GHz CPU, 1.7GB RAM, giving different types of instances[21]. The ECU then maps MIPS for consistency with CloudSim (Table 5), and we assume that every instance needs at least 1 ECU and 1 vCPU (core), as shown in Table 5. The speed of each instance (in terms of MIPS) is the product of number of ECUs (1 ECU = 1GHz) and vCPUs (cores). For example, the speed of a m3.medium instance in Table 5 is 3 (ECUs) X 1 (vCPU) = 3 (GHz).

Instance type	No of vCPUs	No of ECUs	Speed/MIPS (MHz)	Memory (GB)	Storage (GB)	Reserved price (1 yr) ($/hour US East - N. Virginia)
t2.nano	1	1	1000	0.5	1	0.006
t1.micro	1	1	1000	0.613	1	0.02
t2.micro	1	1	1000	1	1	0.013
m1.small	1	1	1000	1.7	160	0.044
m1.medium	1	2	2000	3.75	410	0.087
m3.medium	1	3	3000	3.75	4	0.067

Table 5: Amazon different instance types and their characteristics[16]

Different allocation policies including RR, R, Brs [50], Mpd [51], Lago [50], Carlo [52], FF [46] and FillUp [14] [as described in Sec. 5] combined with different migration approaches

[20] https://www.spec.org/power_ssj2008/

[21] http://www.ec2instances.info

are used to study the effects (trade-off) of energy and performance (hence cost) when different types of workload are taken into account. The migration policies are: (i) No - no migration; (ii) All - migrate all VMs without considering hosts efficiencies or performance; (iii) Perf - migrate for performance only; (iv) Cmcr - migrate relatively long-running VMs to most efficient hosts for energy efficiency [14]; and (v) Perf+Cmcr - migrate relatively long-running VMs to most efficient hosts if and only if better performance is guaranteed.

6.2 Experimental Results

The simulated infrastructure is composed of 12,583 hosts with configuration shown in Table 4. We run the simulation with three types of workload (Povray, Namd, Stream) mapped from the Google trace. The hosts are heterogeneous, and one consequence of this is that the workload could run faster or slower on a different host. Every 5 minute, the migration policy checks for consolidation opportunities, and selects VMs suitable for migration according to the migration policy. For example, Cmcr looks to migrate VMs running for longer times from a list of migration possibilities. Each experiment was performed with five different values for past VM runtime given in minutes [0, 15, 30, 45, and 60], where 0 means migrate all, 15 means migrate only those VMs which are running for 15 minutes or longer, 30 means running for 30 minutes, and so on. Cmcr migrates VMs if the target host is more energy efficient than the source host. However, this ignores performance, and so we add a performance aware migration approach "Perf" to migrate VMs to target hosts only if this will guarantee better, or at least, the expected level of performance.
We simulated three different kinds of workload on a cluster of heterogeneous hosts as discussed in the start of this section. The experiments were performed (repeatedly) using different VM allocation and migration policies with five different values for VMs past runtime (R_{past}) at the time when they were considered for migration. Note that all the results discussed here are the best (minimal) that we achieved. The results for different VM allocation and migration approaches are shown in Table 6 (Povray), Table 7 (Namd) and Table 8 (Stream). The most important columns i.e. energy, performance and ERP are coloured (cyan) to make them more visible. Furthermore, the best approaches are shown in bold face. Similarly, migration statistics and cost recovery details are given in Table 9. The cost savings (%) are calculated based on the amount of energy consumed and service providers' revenue (user cost) with respect to the baseline "no migration" approach. In next sections, we characterize the results for each kind of workload.

GRATIS (0) Workload: In Google data, the Gratis workload runs on 156,886 VMs of five different types [t2.nano = 31,265, t1.micro = 8,777, t2.micro = 30,777, m1.small = 54,873, m1.medium = 31,194], and the results obtained are shown in Table 6. The execution time (performance) varies between 292.73 and 324.86 minutes, and energy consumption varies between 46,461.1 and 50,441.78 kWh, for all allocation and migration policies. The H value corresponds to a combined (single) value of both energy consumption and performance (the inverse of runtime). Note that all execution times are averages. As the RR approach creates more opportunities for consolidation, VMs are fortunate enough to be migrated to better performance hosts, and so jobs will finish faster. Note that the All migration approach is not necessarily energy efficient, and nor does it guarantee good performance. However, Perf migration approach guarantees good performance at the cost of a small increase in energy consumption (less energy efficiency). When there are no migrations, the Gratis workload would complete, on average, in 315.48 minutes. With random (R) allocation policy, although performance is better than the RR approach i.e. 292.73 minutes, however, the energy consumption (49,485.43 kWh) is not the lowest (minimal). The RR approach runs the workload with minimum energy consumption (46,461.1 kWh), and completes the job in 296.58 minutes. If there are no migrations, the FillUp approach beats all other scheduling algorithms (as expected) because it utilizes more energy efficient hosts first.
For Cmcr, the energy usage could be increased if the target host performs worse as the extra time needed may offset any perceived (host) energy efficiency. The other scheduling techniques with exemption of RR, R and Carlo, migrate VMs occasionally (as they initially pack VMs

Scheduling approach	Consolidation technique	R_{past} (minutes)	H_{used} (avg)	DC_{util} (%)	D (MWh)	Energy (MWh)	R_{avg} (hours)	Migrations	**ERP** ($W \times h$)	(%) Cost savings	H
RR	No	-	5,656	18.93	40.015	48.02	315.48	0	3.507	0	0.504
	All	-	1,665	59.34	40.952	49.14	314.45	137,395	3.577	-2.34	0.675
	Perf	-	2,654	40.02	38.913	46.7	300.91	83,751	3.253	2.75	0.096
	Cmcr	30	2,868	37.7	38.718	46.46	296.58	62,276	3.19	3.24	0.001
	Perf+Cmcr	30	3,574	28.91	38.945	46.73	297.83	41,112	3.222	2.67	0.096
R	NO	-	5,424	19.51	39.95	47.94	315.48	0	3.501	0	0.488
	All	-	1,628	57.61	42.035	50.44	311.34	140,437	3.635	-5.22	0.735
	Perf	-	2,479	42.28	38.875	46.65	300.22	101,978	3.242	2.69	0.079
	Cmcr	60	3,340	30.58	**38.84**	49.49	**292.73**	51,538	3.158	2.78	0.006
	Perf+Cmcr	15	3,027	34.27	38.966	46.75	301.73	53,136	3.266	2.46	0.119
Brs	NO	-	3,180	34.02	39.293	47.15	315.48	0	3.443	0	0.279
	All	-	1,377	77.67	38.8	46.56	314.43	8,697	3.389	1.25	0.049
	Perf	-	1,383	77.05	38.783	46.54	314.7	2,753	3.39	1.3	0.039
	Cmcr	60	1,376	77.49	38.781	46.56	314.47	4,061	3.388	1.3	0.037
	Perf+Cmcr	30	1,384	77.04	38.783	46.54	314.73	2,526	3.391	1.3	0.039
Mpd	NO	-	4,312	24.36	39.522	47.43	315.48	0	3.463	0	0.362
	All	-	1,476	68.92	40.554	48.67	317.72	56,898	3.579	-2.61	0.647
	Perf	-	1,694	64.03	38.763	46.52	312.93	22,024	3.37	1.92	0.028
	Cmcr	0	2,115	49.94	39.168	49.26	309.34	15,738	3.366	0.9	0.216
	Perf+Cmcr	15	2,245	48.98	38.779	46.54	307.15	16,736	3.309	1.88	0.036
FF	NO	-	3,189	33.94	39.296	47.16	315.48	0	3.444	0	0.328
	All	-	1,282	75.91	41.562	49.88	314.56	7,355	3.632	-5.77	0.759
	Perf	-	1,285	75.74	41.555	49.87	314.63	4,542	3.632	-5.75	0.759
	Cmcr	45	1,385	77.24	38.786	46.56	314.61	3,242	3.39	1.3	0.04
	Perf+Cmcr	60	1,388	76.92	38.786	46.55	314.67	2,503	3.39	1.3	0.041
FillUp	NO	-	3,179	34.02	39.292	47.15	315.48	0	3.443	0	0.279
	All	-	1,375	77.8	38.797	46.56	314.38	8,732	3.388	1.26	0.047
	Perf	-	1,381	77.22	38.782	46.54	314.58	3,012	3.389	1.3	0.038
	Cmcr	45	1,376	77.49	38.78	46.56	314.44	4,097	3.387	1.3	0.037
	Perf+Cmcr	**30**	**1,381**	**77.22**	**38.782**	**46.54**	**314.58**	**3,012**	**3.389**	**1.3**	0.038
Carlo	NO	-	5,423	19.52	39.951	47.94	315.48	0	3.501	0	0.488
	All	-	1,739	56.8	40.655	48.79	311.13	141,483	3.514	-1.76	0.579
	Perf	-	2,542	41.39	38.899	46.68	300.82	101,660	3.25	2.63	0.09
	Cmcr	30	2,824	36.78	38.904	48.65	297.73	67,401	3.217	2.62	0.083
	Perf+Cmcr	0	2,853	37.46	38.971	46.77	305.23	61,052	3.304	2.46	0.128
Lago	NO	-	3,794	28.27	39.413	47.3	315.48	0	3.454	0	0.324
	All	-	1,518	67.43	**40.184**	48.22	**324.86**	59,298	3.626	-1.95	0.613
	Perf	-	1,775	60.93	38.79	46.55	313.36	16,446	3.376	1.58	0.043
	Cmcr	60	1,745	59.86	40.377	48.48	313.32	12,338	3.514	0.09	0.562
	Perf+Cmcr	30	1,896	56.49	38.796	46.57	312.24	12,952	3.365	1.57	0.046

Table 6: Experimental results (minimal based on R_{past}) for different scheduling and consolidation approaches with Gratis (0) workload - Povray – number of used hosts and datacenter utilization are averaged at 5 minute intervals [H is the harmonic mean of energy and performance: closest to zero represents better approach]

into a host until it is filled and always attempt to fill gaps created by terminated VMs), and could produce minimal results with Perf+Cmcr approach if relatively long-running VMs are migrated only. For Gratis workload, Cmcr produces minimal results (energy consumed) by migrating VMs that are running for 30 - 45 minutes (for different VM allocation policies) or longer. Migration of VMs running for less than 30 minutes were found costly as they were unable to recover their migration cost. If performance of hosts is taken into account, then the more VMs (relatively long-running) we migrate to performance guaranteed hosts, the more efficient allocation and the lower the value for D. If the workload executes faster, then users should pay less for their provisioned resources and thus the system is more energy and cost-efficient both for providers (energy bills) and for users.

Migration of VMs ensures that workloads (VMs) perform better on target hosts, which also increases the probability that migrated VMs will be able to recover their migration cost. The migration statistics in Table 9 show that with performance aware migration, the majority of the migrated VMs (54.25%) run for a sufficient additional duration to be able to recover their migration cost. However, allocation heuristics like FillUp achieve optimal results by migrating only about 0.02% of the VMs. Overall, performance aware migration is up to 3.24% more cost-efficient than no migration, but other approaches can increase costs by up to 6%. Fig. 4 (left) shows the PPR values of different allocation and migration policies for Gratis workload – where maximum energy efficiency of 289.43 (performance to power ratio) is achievable with the proposed Perf+Cmcr technique. Compared to other scheduling and migration heuristics, the PPR value for our proposed VM allocation and consolidation approach shows a balanced trade-off amongst energy, performance and hence cost.

BATCH (2) Workload: For Batch workload, which runs 282,464 VMs of four different types [t2.nano = 56,199, t1.micro = 16,137, t2.micro = 116,582, m1.small = 93,546], our findings are almost similar to the Gratis workload, and efficient heuristics like FF and FillUp obtain minimal results as shown in Table 7. For Batch workload, the execution time (per-

formance) varies between 71.57 and 85.49 minutes, and energy consumption varies between 39,181.97 and 69,104.78 kWh, for all allocation and migration policies. For such short running workloads, migrations could be even more expensive (due to migration cost that could lead to wasted migration efforts [14]), and PERF+CMCR produces good results with minimum energy consumption (39,181.97 kWh) and average VM runtimes (77.2 minutes). Similar to GRATIS workload, if we migrate more VMs, then the performance (execution time) of the system might improve from 85.49 minutes to 71.57 minutes, however, the energy consumption is increased by $\sim 8.33\%$ i.e. from 39,181.97 kWh to 44,749.58 kWh. Similar trade-off between performance and energy was observed for GRATIS workload as well. For BATCH workload, CMCR produces minimal results (energy consumption) by migrating VMs that are running for 15 minutes or longer. The H metric shows the efficiency of the proposed technique "PERF+CMCR".

Scheduling approach	Consolidation technique	R_{past} (minutes)	H_{used} (avg)	DC_{util} (%)	D (MWh)	Energy (MWh)	R_{avg} (hours)	Migrations	**ERP** ($W \times h$)	(%) Cost savings	H
RR	NO	-	5,301	10.28	34.816	41.78	84.76	0	0.82	0	0.159
	ALL	-	1,032	31.72	57.098	68.52	80.27	232,782	1.273	-64	0.765
	PERF	-	3,308	17.32	33.232	39.88	77.85	67,143	0.719	4.55	0.045
	CMCR	60	3,169	18.13	**37.291**	44.75	**71.57**	37,845	0.708	-2.31	0.013
	PERF+CMCR	0	3,347	16.31	33.542	40.25	77.9	47,389	0.726	3.66	0.067
R	NO	-	4,804	11.19	34.652	41.58	84.76	0	0.816	0	0.148
	ALL	-	1,017	36.1	57.587	69.11	81.83	232,288	1.309	-66.19	0.85
	PERF	-	2,744	22.12	32.652	39.18	77.2	99,765	0.7	5.77	0.001
	CMCR	60	2,792	21.08	35.87	68.85	74.57	41,889	0.743	-3.52	0.163
	PERF+CMCR	15	3,054	17.31	33.396	40.54	78.23	40,523	0.726	3.63	0.057
BRS	NO	-	1,502	37.66	33.738	40.49	84.76	0	0.794	0	0.084
	ALL	-	809	60.3	37.423	44.91	84.21	12,402	0.875	-10.92	0.316
	PERF	-	912	59.77	33.526	40.23	84.3	4,105	0.785	0.63	0.068
	CMCR	30	868	61.16	34.457	41.51	84.17	5,991	0.806	-2.13	0.135
	PERF+CMCR	15	912	59.28	33.515	40.23	84.31	2,476	0.785	0.66	0.067
MPD	NO	-	4,522	12.24	34.246	41.1	84.76	0	0.806	0	0.12
	ALL	-	1,200	46.17	33.297	39.96	84.89	89,641	0.785	2.77	0.051
	PERF	-	1,798	31.28	33.591	40.31	80.87	47,336	0.755	1.91	0.072
	CMCR	0	1,386	39.58	33.227	39.87	82.36	90,680	0.76	2.97	0.045
	PERF+CMCR	0	2,042	28.46	33.34	40.01	80.19	46,537	0.743	2.65	0.053
FF	NO	-	1,484	38.92	33.736	40.48	84.76	0	0.794	0	0.084
	ALL	-	896	63.63	33.557	40.27	84.33	11,490	0.786	0.53	0.07
	PERF	-	897	61.51	33.524	40.23	84.4	5,758	0.786	0.63	0.068
	CMCR	15	891	61.78	33.514	43.13	84.4	4,384	0.786	0.66	0.067
	PERF+CMCR	15	898	60.65	33.51	40.23	84.42	2,728	0.786	0.67	0.067
FILLUP	NO	-	1,484	38.92	33.736	40.48	84.76	0	0.794	0	0.084
	ALL	-	893	63.17	33.551	40.26	84.3	11,397	0.786	0.55	0.07
	PERF	-	903	61.41	33.541	40.25	84.43	7,020	0.787	0.58	0.069
	CMCR	30	891	61.44	33.552	40.26	84.37	3,814	0.785	0.67	0.067
	PERF+CMCR	**15**	**899**	**60.38**	**33.54**	**40.25**	**84.46**	**2,789**	**0.786**	**0.66**	0.067
CARLO	NO	-	4,813	11.17	34.659	41.59	84.76	0	0.816	0	0.149
	ALL	-	1,017	36.74	57.501	69.0	82.35	232,550	1.315	-65.91	0.873
	PERF	-	2,664	22.98	33.589	40.31	77.19	99,352	0.72	3.09	0.069
	CMCR	60	2,804	20.52	57.401	68.88	74.31	41,886	0.735	-2.72	0.15
	PERF+CMCR	45	3,311	16.3	**37.448**	44.94	**77.67**	28,725	0.712	4.79	0.028
LAGO	NO	-	3,366	16.46	33.996	40.8	84.76	0	0.8	0	0.103
	ALL	-	989	48.39	33.381	40.06	85.49	72,208	0.793	1.81	0.057
	PERF	-	1,976	31.48	33.495	40.19	79.08	53,119	0.736	1.47	0.064
	CMCR	0	1,000	48.24	33.277	39.93	85.22	69,629	0.788	2.11	0.049
	PERF+CMCR	30	2,295	27.09	33.537	40.24	79.13	19,286	0.734	1.74	0.058

Table 7: Experimental results (minimal based on R_{past}) for different scheduling and consolidation approaches with BATCH (2) workload - NAMD – number of used hosts and datacenter utilization are averaged at 5 minute intervals [H is the harmonic mean of energy and performance: closest to zero represents better approach]

As shown in Table 9, migration of VMs to better-performance hosts also increases the probability that they will recover their migration cost and subsequently run more efficiently to save energy. The PERF+CMCR migration approach could save approximately 5.77% in costs compared to the baseline, no migration, approach. The H value of PERF+CMCR is between the H values of PERF and CMCR approaches which shows a balanced trade-off between energy consumption and performance. Fig. 4 (middle) shows the PPR values of different allocation and migration policies for BATCH workload – where maximum energy efficiency of 568.41 (performance to power ratio) is achievable with PERF+CMCR. Note that the "migrate all" approach consolidates VMs on fewer hosts, which increases the resource utilization and also improves the performance. However, there is a 39.02% increase in total energy consumption. Therefore, it is not necessary that increase in resource utilization will always decrease the system energy consumption. These findings contradict what researchers [55], [56] are trying to achieve in terms of greater energy efficiency in datacenters through increased resource utilization. More research is required to establish the levels of resource utilization that would increase energy efficiency

without any performance loss when different kinds of workload are taken into account. It seems like this would vary significantly, depending on workload mixtures and how each utilizes the system.

PRODUCTION (9) Workload: An unexpected behaviour is observed for PRODUCTION workload which runs 52,959 VMs of five different types [t2.nano = 10,596, t1.micro = 2,767, t2.micro = 10,305, m1.small = 18,677, m1.medium = 10,614], as shown in Table 8. For PRODUCTION workload, the average runtime (performance) varies between 963.59 and 1,107.42 minutes, and energy consumption varies between 61,957.91 and 63,054.65 kWh, for all allocation and migration policies. Previously, we demonstrated [14], that migrating relatively long-running VMs is more economical and energy efficient. Therefore, we were expecting similar findings for long-running VMs for the PRODUCTION workload. However, we observed that although performance is improved, energy consumption is worse than "no migration" approach. For example, with FILLUP combined with PERF+CMCR approach, performance is improved from 1,107.42 minutes to 1,086.74 minutes. However, energy consumption is increased from 61,957.91 kWh to 63.046.62 kWh. Note the behaviour of H for efficient heuristics like BRS, LAGO, MPD, FF and FILLUP where no migration approach is more energy and performance efficient.

Scheduling approach	Consolidation technique	R_{past} (minutes)	H_{used} (avg)	DC_{util} (%)	D (MWh)	Energy (MWh)	R_{avg} (hours)	Migrations	**ERP** ($W \times h$)	(%) Cost savings	H
RR	NO	-	3,359	22.85	**52.297**	62.76	**1,107.42**	0	16.087	0	0.841
	ALL	-	1,075	64.6	52.052	62.46	1,014.44	57,678	14.668	0.47	0.4
	PERF	-	1,203	55.21	**51.809**	62.17	**963.59**	55,167	13.868	0.93	0.001
	CMCR	60	1,762	40.61	52.06	62.77	1,035.49	41,106	14.974	0.45	0.483
	PERF+CMCR	45	1,856	38.47	51.96	62.57	1,035.44	37,392	14.945	0.64	0.417
R	NO	-	3,238	23.27	52.254	62.7	1,107.42	0	16.074	0	0.808
	ALL	-	1,072	65.94	52.259	62.71	1,047.18	63,170	15.201	-0.01	0.629
	PERF	-	1,187	56.14	51.95	62.34	985.23	64,180	14.217	0.58	0.211
	CMCR	60	1,783	38.57	52.019	62.86	1,027.54	48,810	14.848	0.45	0.434
	PERF+CMCR	60	1,931	35.16	51.937	62.64	1,009.74	45,450	14.567	0.61	0.327
BRS	NO	-	1,763	44.98	51.634	61.96	1,107.42	0	15.883	0	0.005
	ALL	-	937	80.89	52.546	63.06	1,086.9	3,262	15.864	-1.77	0.922
	PERF	-	942	80.53	52.543	63.05	1,087.18	2,612	15.868	-1.77	0.922
	CMCR	60	937	80.87	52.542	63.06	1,086.78	3,123	15.862	-1.76	0.919
	PERF+CMCR	45	946	80.24	52.549	63.05	1,087.92	2,520	15.88	-1.78	0.927
MPD	NO	-	2,163	34.66	51.749	62.1	1,107.42	0	15.919	0	0.228
	ALL	-	1,117	66	51.967	62.36	1,052.55	20,944	15.194	-0.42	0.46
	PERF	-	1,116	64.99	51.913	62.3	1,034.07	17,581	14.912	-0.32	0.378
	CMCR	60	1,223	61.34	52.021	62.51	1,076.32	12,868	15.553	-0.52	0.551
	PERF+CMCR	60	1,300	55.77	52.036	62.48	1,074.26	11,149	15.528	-0.56	0.561
FF	NO	-	1,766	44.91	51.632	61.96	1,107.42	0	15.883	0	0.002
	ALL	-	938	81.02	52.537	63.04	1,085.6	2,989	15.843	-1.75	0.912
	PERF	-	939	80.88	52.535	63.04	1,085.98	2,711	15.848	-1.75	0.913
	CMCR	15	937	80.88	52.534	63.04	1,085.38	2,724	15.839	-1.75	0.91
	PERF+CMCR	15	938	80.78	52.533	63.04	1,085.45	2,694	15.84	-1.75	0.91
FILLUP	NO	-	1,766	44.91	51.632	61.96	1,107.42	0	15.883	0	0.002
	ALL	-	939	81.06	52.542	63.05	1,086.8	3,072	15.862	-1.76	0.92
	PERF	-	940	80.46	52.538	63.05	1,086.68	2,522	15.859	-1.75	0.917
	CMCR	60	939	80.93	52.542	63.05	1,086.62	2,663	15.858	-1.75	0.917
	PERF+CMCR	**60**	**941**	**80.43**	**52.539**	**63.05**	**1,086.74**	**2,423**	**15.86**	**-1.76**	0.918
CARLO	NO	-	3,234	23.28	52.25	62.7	1,107.42	0	16.073	0	0.806
	ALL	-	1,123	62.99	52.067	62.48	1,045.82	64,443	15.126	0.35	0.519
	PERF	-	1,229	54.54	51.834	62.2	991.71	65,209	14.279	0.8	0.208
	CMCR	60	11,815	37.82	51.867	62.61	1,024.31	49,647	14.758	0.73	0.32
	PERF+CMCR	60	1,968	34.4	51.819	62.43	1,006.29	46,405	14.485	0.82	0.243
LAGO	NO	-	2,117	35.32	51.737	62.09	1,107.42	0	15.915	0	0.208
	ALL	-	1,109	68.48	52.029	62.44	1,079.64	18,363	15.604	-0.56	0.564
	PERF	-	1,183	61.28	52.048	62.46	1,076.94	13,381	15.57	-0.6	0.576
	CMCR	60	1,205	62.02	52.025	62.48	1,075.24	12,477	15.539	-0.56	0.553
	PERF+CMCR	30	1,246	58.11	52.028	62.49	1,073.76	11,403	15.518	-0.56	0.553

Table 8: Experimental results (minimal based on R_{past}) for different scheduling and consolidation approaches with PRODUCTION (9) workload - STREAM – number of used hosts and datacenter utilization are averaged at 5 minute intervals [H is the harmonic mean of energy and performance: closest to zero represents better approach]

The results repeat similar findings (as discussed above for GRATIS and BATCH workloads) that "migrate all" approach is more expensive in terms of improved performance and decreased energy efficiency. The PRODUCTION workload runs for longer and, for randomized allocation heuristics, significant energy could be saved that can be up to ∼0.93% as compared to the "no migration" approach, if the workload is migrated to better performing hosts. Performance can be improved from 1,107.42 minutes to 963.59 minutes when performance aware migrations are combined with inefficient RR and R allocation policies. We were expecting similar sav-

ings using PERF+CMCR which migrate relatively long-running VMs to more energy efficient hosts. However, we observed that PERF+CMCR approach is less energy and cost-efficient for PRODUCTION workload. There are three possible reasons for this unexpected behaviour: *(i)* energy efficient hosts do not always guarantee better performance; *(ii)* if performance is worse on an energy efficient host, the savings earned by host efficiency, are less than the additional energy consumption due to increase in runtime (slow performance); and *(iii)* if *(ii)* happens for a longer time, more energy is consumed.

Scheduling approach	RR				R				BRS				MPD				Workload type
	dc	perf	cmcr	p+c	dc	perf	cmcr	p+c	dc	perf	cmcr	p+c	dc	perf	cmcr	p+c	
Migratable VMs (%)	0.88	0.53	0.4	0.26	0.9	0.65	0.33	0.34	0.06	0.02	0.03	0.02	0.36	0.14	0.1	0.11	GRATIS
VMs recovered	35.07	46.08	39.24	49.63	35.03	44.47	38.38	48.81	11.89	7.95	11.25	8.14	34.01	46.59	50.27	54.25	
$Cost_m$ (%)	0.82	0.24	0.13	0.17	0.82	0.35	0.15	0.14	0.04	0.01	0.02	0.01	0.32	0.17	0.32	0.16	BATCH
	14.58	39.33	26.24	33.89	15.57	34.58	21.26	25.82	12.47	8.6	9	7.05	19.84	21.96	27.09	30.71	
	1.09	1.04	0.78	0.71	1.19	1.21	0.92	0.86	0.06	0.05	0.06	0.05	0.4	0.33	0.24	0.21	PRODUCTION
	70.43	67.59	68.04	67.44	74.14	72.26	70.11	69.09	18.61	10.18	15.89	9.1	62.19	64.31	54.9	51.34	
Scheduling approach	FF				FILLUP				CARLO				LAGO				Workload type
	dc	perf	cmcr	p+c	dc	perf	cmcr	p+c	dc	perf	cmcr	p+c	dc	perf	cmcr	p+c	
Migratable VMs (%)	0.05	0.03	0.02	0.02	0.06	0.02	0.03	0.02	0.9	0.65	0.43	0.39	0.38	0.1	0.08	0.08	GRATIS
VMs recovered	5.64	5.77	4.24	4.5	6.83	4.26	10.99	10.17	35.49	44.68	38.48	48.22	35.09	39.56	39.41	25.32	
$Cost_m$ (%)	0.04	0.02	0.02	0.01	0.04	0.02	0.01	0.01	0.82	0.35	0.15	0.1	0.26	0.19	0.25	0.07	BATCH
	7.67	5.42	6.34	3.8	8.11	9.33	9.98	8.21	15.45	34.13	20.84	26.06	22.12	14.5	23.53	9.21	
	0.06	0.05	0.05	0.05	0.06	0.05	0.05	0.05	1.22	1.23	0.94	0.88	0.35	0.25	0.24	0.22	PRODUCTION
	6.79	6.49	3.3	4.1	12.37	6.98	15.55	9.02	74.27	72.77	70.09	69.58	60.59	63.48	54.1	50.92	

Table 9: Cost recovery and proportion (%) of migratable VMs using different scheduling and consolidation techniques for GRATIS (0), BATCH (2) and PRODUCTION (9) workloads [using PERF with CMCR (p+c) enables more VMs to recover their migration cost]

Largely, unlike GRATIS and BATCH workload, the migrated VMs recovered their migration cost due to their long-running behaviour, as shown in Table 9. However, the extra energy cost as observed for the PRODUCTION workload is due to the long-running behaviour of VMs. If VMs runs for longer on a less energy efficient and/or low performance host, there will be more energy usage due to increase in execution time (performance). Fig. 4 (right) shows the PPR values of different allocation and migration policies for PRODUCTION workload – where maximum energy efficiency of 78.2 (performance to power ratio) is achievable with performance-aware consolidation technique.

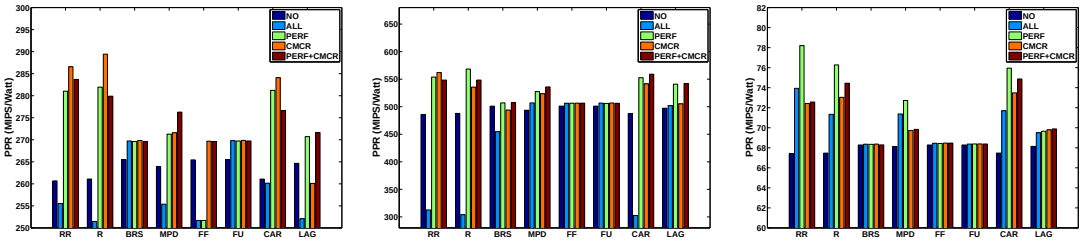

Fig. 4: Performance to power ratio (PPR) for different allocation & migration policies and workloads – GRATIS, BATCH and PRODUCTION (from left to right) [PERF+CMCR balances the trade-off between energy, performance and cost]

For BATCH workload, the maximum energy efficiency was achieved by migrating VMs that have run for 15 minutes or longer. Similarly, for GRATIS workload, this value was observed as either 30 or 45 (for different heuristics). For long-running VMs (PRODUCTION workload),

migrating VMs running for 1 hour or longer is more cost-energy efficient. For PRODUCTION workload, we have only considered migration of those VMs that were running for one hour or shorter. However, it is possible that more energy could be saved if VMs running for longer than 1 hour, are migrated, particularly, to target hosts that offer better performance and energy efficiency than the source hosts [14].

6.3 Results Discussion

For each scheduling approach, each VM requests cloud resources (CPU, memory) and comes with a host ID to accommodate the VM on that host (according to Google data as explained in Sec. 3). Therefore, for each policy, when migrations are not considered, execution time is similar (or same). However, the number of average hosts in use (5 minute intervals) and datacenter utilization (%) confirms the efficiency of the FILLUP VM allocation approach – which provides consistency with our previous findings [14]. For GRATIS workload, the RR approach creates more opportunities for the migration, hence, if more VMs are migrated to better performing hosts, performance can be improved. The other scheduling techniques, except RR, R and CARLO, migrate VMs occasionally and might produce minimal results if each VM is allocated to a host that performs better. If performance of hosts is taken into account, then the more VMs (relatively long-running) we migrate to performance guaranteed hosts, the more efficient allocation and the lower the value for D.
Similar behaviour is also observed for BATCH workload where efficient allocation heuristics like FF and FILLUP obtain the minimal results in terms of energy used and number of migrations. For such short running workloads, migrations could be expensive, and PERF+CMCR produces good results with minimum energy consumption and average per VM runtime duration. However, for the PRODUCTION workload, which runs for longer durations, we observed an unexpected behaviour. The per VM average runtime duration is proportional to the user monetary cost – the longer the VM runs, the more it will cost. For different allocation heuristics and different kinds of workload, the performance-aware migration technique could save up-to 2.65% energy as compared to "no migration" approach. These savings can be up to 3.66% if CMCR is also considered combined with performance-aware migration as shown in Table 8. Considering a PUE of 1.2 and energy price at $0.08 per kWh, the proposed performance-aware migration technique would save $260 per day, which is approximately a $0.1m saving over a year. Similar numbers for energy and cost savings are also achievable for BATCH workload. The BATCH workload runs for shorter duration and the savings could be up to $1988.96 per day, making a total savings of ∼$0.73m annually – for similar datacenter configurations. The PRODUCTION workload runs for longer durations and for similar datacenter set-up, a monthly savings of ∼$0.02m are achievable at the rate of $44.26 per day. These saving for GRATIS (0), BATCH (2) and PRODUCTION (9) workloads compare favourably to a maximum projected usage of the same 12,583 hosts cluster of approximately $2.4m/year.
The results suggest that reducing the system energy usage affects the compute performance. CMCR [14] tries to migrate only relatively long-running VMs to more energy efficient hosts, that might not perform better for a particular workload type. In such situations the VMs run for a longer duration, and result in more energy consumption. Similar trade-offs between system energy consumption and performance variations are also observed in [57]. For GRATIS workload, CMCR produces minimal results by migrating VMs that run for 15 to 30 minutes or longer. Similar long-running VMs were also most efficient for BATCH workload, however, in PRODUCTION workload, the past runtime R_{past} of migrated VMs is 45 minutes to 1 hour. For these long-running VMs, the results show that there are further opportunities for greater energy efficiency and performance, therefore, user cost, if VMs of $R_{past} > 1.5$ hours are only migrated. In all results, as shown in Tables 6, 7, 8, a column group is shown for each of the examined metrics -– energy and cost used by hosts, average number of hosts used (H_{used}), average datacenter utilization (%), VM average runtime (R_{avg}), the service revenue obtained from hosting the VMs (user cost %), total cost savings in US dollars (%) and H (combined single value for energy and performance) for each kind of workload. Further, we categorize our observations in the remaining part of this section.

Generalization: Based on our findings, we generalize the three different types of workload benchmarks w.r.t the continuing trade-off between overall system energy consumption, performance and total cost, that we observed for different allocation and migration policies. We validated and verified our findings through running the experiments several times, with different experimental assumptions and parameters; and determined whether the generalization is correct or not. We observed a consistency among the major findings, with variability in energy consumption and performance (hence cost). Migration of VMs to better performance guaranteed hosts increases the performance (that reduces user per hour cost), however, it is also possible that energy consumption is increased. Thus, a host may guarantee better performance for a specific kind of workload, but may be at cost of more energy usage. We observed that the proposed performance-aware migration technique improves the workload performance (runtime), and efficient allocation heuristics (FillUp) reduces the energy consumption. Note that performance and cost gives almost the same values when migrations are not considered because initially all VMs are placed on similar hosts.

Resource Utilization: Another interesting point that we observed is about the resource utilization. Largely, the literature [Sec. 2] asserts that increasing resource utilization could decrease energy consumption. However, higher resource utilization could increase energy consumption. This affect can be related either to Jevons Paradox[22] (*the easier you make it to consume the product the greater the consumption will be*), *race-to-idle* and/or *never-idle* heuristics [34]. When peak performance is not needed then operating the CPU at lower utilization level enables the resource/energy management technique to save more energy by dynamically ratcheting down the CPU power states through DVFS (Dynamic Voltage and Frequency Scaling). However, if the CPU operates near maximum capacity (higher utilization) most of the time, this technique would offer little advantage[23]. Therefore, several techniques like [58], demonstrated greater energy efficiency with two utilization thresholds for a resource (CPU); (i) a lower threshold and (ii) an upper threshold.

Initial Placement vs. Migration Placement: Lastly, we repeatedly performed the experiments [Sec. 6.1] to demonstrate that how two different policies; one for initial VM placement and one for migration placement - would affect the energy consumption for different kinds of workload. This will help providers to know either a single policy or two different policies for both placement decisions are more economical and energy efficient. Table 10 shows the eight possibilities for every initial allocation policy combined with all eight migration placement policies. The three kinds of workload i.e. Gratis, Batch and Production are represented by G, B and P respectively. The FillUp migration placement policy tries to accommodate the migrated VM on a more energy efficient host and produces minimal results (energy usage) when combined with all scheduling approaches, dependent on the workload type.

Alloc. policy	Migration placement techniques							
	RR	R	Brs	Mpd	FF	FillUp	Carlo	Lago
RR	P				B	G		
R					G,B	P		
Brs			G		P	B		
Mpd					B,P	G		
FF					B	G,P		
FillUp						G,B,P		
Carlo				B		P	G	
Lago					G	P		B

Table 10: Initial scheduling policies vs migration placement policies for Gratis-G, Batch-B and Production-P workloads - [left side denotes initial placement policies and top row represents migration placement policies]

[22] https://en.wikipedia.org/wiki/Jevons_paradox

[23] http://www.buildings.com/article-details/articleid/6000/title/10-ways-to-save-energy-in-your-data-center

Comparison to other approaches: We compare the EPC-AWARE migration technique to three well-known migration techniques i.e. KHANNA [23], SANDPIPER [29] and PMAPPER (FFD) [22] in terms of energy consumption and performance. We made slight modification to the implementation of these policies in order to account for the migration energy cost [14]. The experiments, that run for 12 hours, were performed on the PRODUCTION workload with the same performance parameters and datacenter set-up (which consists of 3,000 hosts) as discussed in Sec. 6.1 and Sec. 6.2. SANDPIPER performs better and EPC-AWARE is more energy efficient than other approaches, as shown in Table. 11. Note that the "tasks completed" is the total number of VMs that were able to finish their execution during the experiment. Furthermore, FILLUP allocation, which is used in EPC-AWARE migration technique, is more energy efficient than the FFD allocation, which is used in PMAPPER.

Policy	Energy (kWh)	Avg. runtime (minutes)	Tasks completed	Number of migrations	Max. hosts used
KHANNA	3,860.26	426.57	1,282	10,376	1,147
SANDPIPER	4,011.54	384.97	1,589	7,812	1,249
EPC-AWARE	3,858.41	417.02	1,336	0	1,070
PMAPPER [FFD]	3,918.83	417.02	1,336	0	1,240

Table 11: Comparing EPC-AWARE with KHANNA, SANDPIPER and PMAPPER [FFD]

Analysis of the proposed algorithm: The proposed EPC-AWARE migration approach is based on the "FILLUP" approach presented in [14]. The "FILLUP" allocation policy is based on the First Fit (FF) approach that is demonstrated to be not using more than twice bins (hosts) as used by an optimal (OPT) solution [59]. It means that only after the bin fills with more than $\frac{V}{2}$ items or if an item (VM) with a size larger than $\frac{V}{2}$ arrives, the algorithm may start (switched on) a new host. Thus, if we have a set of H hosts, then, at least, $(H-1)$ hosts can be more than half full. Now, if we consider an improved version of the FF heuristic i.e. FFD (First Fit decreasing), where hosts are sorted in decreasing order of their sizes and which is guaranteed to be not using more than 1.22×OPT hosts [59]. In "FILLUP" heuristic, hosts are sorted based on their sizes and energy consumption, therefore, the total number of hosts used by the "FILLUP" approach are given by:

$$1.22 \times OPT \leq \text{FILLUP} \leq 2 \times OPT \tag{17}$$

If a VM is allocated to a suitable host in the first iteration (i.e. the best case), the "FILLUP" approach time complexity is $\mathcal{O}(1)$. In the worst case, it is possible that all (n) hosts are scanned, therefore, the time taken in allocation can be up to $\mathcal{O}(n)$. For the migration policies, this (worst case) can be up to $\mathcal{O}(mn)$ and $\mathcal{O}(1)$ in the best case; where m is the total number of VMs selected for migration.

7 Conclusions and Future Work

Variable performance of similar instances (VMs) results in a different amount of work that these instances can complete per unit of billable time [18]. Hence, for some fixed amount of work, variability in performance would produce different workload execution time, and so potentially increase costs. This led the researchers to propose various strategies to exploit such performance variation in clouds and find instances that would perform better at the same price. In this paper, we discussed the performance, cost and energy implications for different kinds of workload, along with suggesting a VM allocation and reallocation technique for improving performance and energy (hence cost) conservation in a datacenter. Based on our extensive simulations we generalized three different types of workload benchmark for a similar trade-off among performance, energy and cost. We observed that the performance of different scheduling and migration strategies differ widely.

For heterogeneous workloads and clouds, this paper evaluates different scheduling and migration techniques that demonstrate a diversity in energy efficiency and performance (hence cost)

trade-off spaces. Some policies improve energy efficiency as it migrates (VMs) to higher performance guaranteed hosts while the other experiences a reduction in energy efficiency as the system performance increases. Applying the proposed Energy-Performance-Cost (EPC) aware migration heuristic achieves near minimal (balanced) energy consumption, improves performance and reduces the user cost. Different resource allocation and migration strategies are clearly necessary for cloud platforms exhibiting variation in performance in order to attain higher energy efficiency while meeting performance targets. It is also reasonable to expect that, even on a single platform, different kinds of workload may have different energy and performance trade-off spaces [34]. Therefore, a single allocation heuristic may not be suitable for all applications (workloads) on that platform to achieve greater energy efficiency and performance improvements. Furthermore, adding more of the best (VMs) to the same (host) in order to maximize overall performance would be bad for it in terms of co-location and performance loss [Sec. 3]. And, indeed, this is a flaw of the present work that future work would have to address.

Note that different cases simulated in this work does not include the networking cost as well as latencies (delays) which can be a constraint for the overall performance. Furthermore, SLA violations and the SLA penalties, are also not considered in the cost model which could impact the overall cost of scheduling and migration techniques. A future implementation of this work may consider these costs as well as performance-aware VM allocation policies – instance seeking algorithms.

Acknowledgement This work is supported by Department of Computer Science, University of Surrey, UK and Abdul Wali Khan University, Mardan, Pakistan.

References

1. A Shehabi, SJ Smith, N Horner, I Azevedo, R Brown, J Koomey, E Masanet, D Sartor, M Herrlin, and W Lintner. United states data center energy usage report. *Lawrence Berkeley National Laboratory, Berkeley, California. LBNL-1005775 Page*, 4, 2016.
2. Tiago C. Ferreto, Marco A S Netto, Rodrigo N. Calheiros, and César A F De Rose. Server consolidation with migration control for virtualized data centers. *Future Generation Computer Systems*, 27(8):1027–1034, 2011.
3. Accenture. Smarter2030: Ict solutions for 21st century challenges online:. http://smarter2030.gesi.org/downloads/Full_report2.pdf, 2015.
4. Qi Zhang, Mohamed Faten Zhani, Shuo Zhang, Quanyan Zhu, Raouf Boutaba, and Joseph L Hellerstein. Dynamic energy-aware capacity provisioning for cloud computing environments. In *Proceedings of the 9th international conference on Autonomic computing*, pages 145–154. ACM, 2012.
5. http://www.telegraph.co.uk/finance/newsbysector/energy/11923465/Blackout-risk-rises-as-UK-energy-crisis-deepens.html. [Online; accessed 21-July-16].
6. techUK. Data centres and power: Fact or fiction? http://www.techuk.org/insights/reports/item/275-data-centres-and-power-fact-or-fiction. [Online; accessed 23-Aug-16].
7. John O'Loughlin and Lee Gillam. Sibling virtual machine co-location confirmation and avoidance tactics for public infrastructure clouds. *The Journal of Supercomputing*, 72(3):961–984, 2016.
8. John OLoughlin and Lee Gillam. Towards performance prediction for public infrastructure clouds: An ec2 case study. In *Cloud Computing Technology and Science (CloudCom), 2013 IEEE 5th International Conference on*, volume 1, pages 475–480. IEEE, 2013.
9. John O'Loughlin and Lee Gillam. Performance evaluation for cost-efficient public infrastructure cloud use. In *International Conference on Grid Economics and Business Models*, pages 133–145. Springer, 2014.
10. Robert Basmadjian, Florian Niedermeier, and Hermann De Meer. Modelling and analysing the power consumption of idle servers. In *Sustainable Internet and ICT for Sustainability (SustainIT), 2012*, pages 1–9. IEEE, 2012.
11. David Meisner, Brian T Gold, and Thomas F Wenisch. Powernap: eliminating server idle power. In *ACM Sigplan Notices*, volume 44, pages 205–216, 2009.
12. Dimitris Tsirogiannis, Stavros Harizopoulos, and Mehul A Shah. Analyzing the energy efficiency of a database server. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 231–242. ACM, 2010.

13. NRDC. America's Data Centers Are Wasting Huge Amounts of Energy: critical action needed to save billions of dollars and kilowatts. *NRDC Issue Brief, IB:14-08-A*, pages 1–6, 2014.
14. Muhammad Zakarya and Lee Gillam. An energy aware cost recovery approach for virtual machine migration. In *Economics of Grids, Clouds, Systems, and Services - 13th International Conference, GECON 2016, Athens, Greece, September 20-22, 2016, Revised Selected Papers*, pages 175–190, 2016.
15. Charles Reiss, John Wilkes, and Joseph L Hellerstein. Google cluster-usage traces: format+ schema. *Google Inc., Mountain View, CA, USA, Technical Report*, 2011.
16. Rodrigo N. Calheiros, Rajiv Ranjan, Anton Beloglazov, César A F De Rose, and Rajkumar Buyya. CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software - Practice and Experience*, 41(1):23–50, 2011.
17. Fei Xu, Fangming Liu, Hai Jin, and Athanasios V Vasilakos. Managing performance overhead of virtual machines in cloud computing: A survey, state of the art, and future directions. *Proceedings of the IEEE*, 102(1):11–31, 2014.
18. John O'Loughlin and Lee Gillam. Re-appraising instance seeking in public clouds. In *Science and Information Conference (SAI), 2015*, pages 807–815. IEEE, 2015.
19. Michael Armbrust, Armando Fox, Rean Griffith, Anthony D Joseph, Randy H Katz, Andrew Konwinski, Gunho Lee, David A Patterson, Ariel Rabkin, Ion Stoica, et al. Above the clouds: A berkeley view of cloud computing. 2009.
20. Fei Xu, Fangming Liu, and Hai Jin. Heterogeneity and interference-aware virtual machine provisioning for predictable performance in the cloud. *IEEE Transactions on Computers*, 65(8):2470–2483, 2016.
21. Fei Xu, Fangming Liu, Linghui Liu, Hai Jin, Bo Li, and Baochun Li. iaware: Making live migration of virtual machines interference-aware in the cloud. *IEEE Transactions on Computers*, 63(12):3012–3025, 2014.
22. Akshat Verma, Puneet Ahuja, and Anindya Neogi. pMapper: Power and migration cost aware application placement in virtualized systems. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 5346 LNCS, pages 243–264, 2008.
23. G Khanna, K Beaty, G Kar, and a Kochut. Application Performance Management in Virtualized Server Environments. *2006 IEEEIFIP Network Operations and Management Symposium NOMS 2006*, 20(D):373–381, 2006.
24. George Kousiouris, Tommaso Cucinotta, and Theodora Varvarigou. The effects of scheduling, workload type and consolidation scenarios on virtual machine performance and their prediction through optimized artificial neural networks. *Journal of Systems and Software*, 84(8):1270–1291, 2011.
25. Draen Lucanin, Ilia Pietri, Ivona Brandic, and Rizos Sakellariou. A cloud controller for performance-based pricing. In *2015 IEEE 8th International Conference on Cloud Computing*, pages 155–162. IEEE, 2015.
26. Hao Zhuang, Xin Liu, Zhonghong Ou, and Karl Aberer. Impact of instance seeking strategies on resource allocation in cloud data centers. In *IEEE CLOUD*, pages 27–34, 2013.
27. Seung-Hwan Lim. *Managing performance and energy in large scale data centers*. PhD thesis, The Pennsylvania State University, 2012.
28. Zhuoyao Zhang, Ludmila Cherkasova, and Boon Thau Loo. Exploiting cloud heterogeneity to optimize performance and cost of mapreduce processing. *ACM SIGMETRICS Performance Evaluation Review*, 42(4):38–50, 2015.
29. Timothy Wood, Prashant Shenoy, Arun Venkataramani, and Mazin Yousif. Sandpiper: Black-box and gray-box resource management for virtual machines. *Computer Networks*, 53(17):2923–2938, 2009.
30. Can Hankendi and Ayse K Coskun. Energy-efficient server consolidation for multi-threaded applications in the cloud. In *Green Computing Conference (IGCC), 2013 International*, pages 1–8. IEEE, 2013.
31. Gueyoung Jung, Matti A Hiltunen, Kaustubh R Joshi, Richard D Schlichting, and Calton Pu. Mistral: Dynamically managing power, performance, and adaptation cost in cloud infrastructures. In *Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference on*, pages 62–73. IEEE, 2010.
32. Qi Zhang, Mohamed Faten Zhani, Raouf Boutaba, and Joseph L Hellerstein. Harmony: Dynamic heterogeneity-aware resource provisioning in the cloud. In *Distributed Computing Systems (ICDCS), 2013 IEEE 33rd International Conference on*, pages 510–519. IEEE, 2013.
33. Karim Djemame, Richard E. Kavanagh, Django Armstrong, Francesc Lordan, Jorge Ejarque, Mario Macías, Raúl Sirvent, Jordi Guitart, and Rosa M. Badia. Energy efficiency support through

intra-layer cloud stack adaptation. In *Economics of Grids, Clouds, Systems, and Services - 13th International Conference, GECON 2016, Athens, Greece, September 20-22, 2016, Revised Selected Papers*, pages 129–143, 2016.

34. Connor Imes and Henry Hoffmann. Minimizing energy under performance constraints on embedded platforms: resource allocation heuristics for homogeneous and single-isa heterogeneous multi-cores. *ACM SIGBED Review*, 11(4):49–54, 2015.
35. Chung-Hsing Hsu, Jeffery A Kuehn, and Stephen W Poole. Towards efficient supercomputing: searching for the right efficiency metric. In *Proceedings of the 3rd ACM/SPEC International Conference on Performance Engineering*, pages 157–162. ACM, 2012.
36. Kejiang Ye, Zhaohui Wu, Chen Wang, Bing Bing Zhou, Weisheng Si, Xiaohong Jiang, and Albert Y Zomaya. Profiling-based workload consolidation and migration in virtualized data centers. *IEEE Transactions on Parallel and Distributed Systems*, 26(3):878–890, 2015.
37. Joydeep Mukherjee, Diwakar Krishnamurthy, and Jerry Rolia. Resource contention detection in virtualized environments. *IEEE Transactions on Network and Service Management*, 12(2):217–231, 2015.
38. Luiz André Barroso and Urs Hölzle. The case for energy-proportional computing. *Computer*, 40(12):33–37, 2007.
39. Albert Greenberg, James Hamilton, David A Maltz, and Parveen Patel. The cost of a cloud: research problems in data center networks. *ACM SIGCOMM computer communication review*, 39(1):68–73, 2008.
40. Varun Gupta. *Stochastic models and analysis for resource management in server farms*. PhD thesis, Intel Corporation, 2011.
41. Anshul Gandhi, Varun Gupta, Mor Harchol-Balter, and Michael Kozuch. Energy-efficient dynamic capacity provisioning in server farms. *School of Computer Science, Carnegie Mellon University, Tech. Rep. CMU-CS-10-108*, 2010.
42. Charles Reiss, Alexey Tumanov, Gregory R. Ganger, Randy H. Katz, and Michael a. Kozuch. Heterogeneity and dynamicity of clouds at scale. *Proceedings of the Third ACM Symposium on Cloud Computing - SoCC '12*, pages 1–13, 2012.
43. Norman Bobroff, Andrzej Kochut, and Kirk Beaty. Dynamic placement of virtual machines for managing sla violations. In *2007 10th IFIP/IEEE International Symposium on Integrated Network Management*, pages 119–128. IEEE, 2007.
44. Charles Reiss, a Tumanov, and Gr Ganger. Towards understanding heterogeneous clouds at scale: Google trace analysis. *... Center for Cloud ...*, 2012.
45. John O'Loughlin and Lee Gillam. Should infrastructure clouds be priced entirely on performance? an EC2 case study. *IJBDI*, 1(4):215–229, 2014.
46. Anton Beloglazov, Jemal Abawajy, and Rajkumar Buyya. Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing. *Future Generation Computer Systems*, 28(5):755–768, 2011.
47. William Voorsluys, James Broberg, Srikumar Venugopal, and Rajkumar Buyya. Cost of virtual machine live migration in clouds: A performance evaluation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 5931 LNCS, pages 254–265, 2009.
48. Vincenzo De Maio, Gabor Kecskemeti, and Radu Prodan. A workload-aware energy model for virtual machine migration. In *2015 IEEE International Conference on Cluster Computing*, pages 274–283. IEEE, 2015.
49. Vincenzo De Maio, Radu Prodan, Shajulin Benedict, and Gabor Kecskemeti. Modelling energy consumption of network transfers and virtual machine migration. *Future Generation Computer Systems*, 56:388–406, 2016.
50. Daniel Guimaraes do Lago, Edmundo RM Madeira, and Luiz Fernando Bittencourt. Power-aware virtual machine scheduling on clouds using active cooling control and dvfs. In *Proceedings of the 9th International Workshop on Middleware for Grids, Clouds and e-Science*, page 2. ACM, 2011.
51. Anton Beloglazov and Rajkumar Buyya. Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints. *IEEE Transactions on Parallel and Distributed Systems*, 24(7):1366–1379, 2013.
52. Carlo Mastroianni, Michela Meo, and Giuseppe Papuzzo. Probabilistic consolidation of virtual machines in self-organizing cloud data centers. *IEEE Transactions on Cloud Computing*, 1(2):215–228, 2013.
53. Rodrigo N Calheiros, Rajiv Ranjan, Anton Beloglazov, César AF De Rose, and Rajkumar Buyya. Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and Experience*, 41(1):23–50, 2011.

54. Carlo Mastroianni, Michela Meo, and Giuseppe Papuzzo. Analysis of a self-organizing algorithm for energy saving in data centers. In *Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2013 IEEE 27th International*, pages 907–914. IEEE, 2013.
55. Anton Beloglazov and Rajkumar Buyya. Energy Efficient Resource Management in Virtualized Cloud Data Centers. In *10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pages 826–831, 2010.
56. Seung-Hwan Lim, Bikash Sharma, Byung Chul Tak, and Chita R Das. A dynamic energy management scheme for multi-tier data centers. In *Performance Analysis of Systems and Software (ISPASS), 2011 IEEE International Symposium on*, pages 257–266. IEEE, 2011.
57. James William Smith and Ian Sommerville. Understanding tradeoffs between power usage and performance in a virtualized environment. In *IEEE CLOUD*, pages 725–731, 2013.
58. Anton Beloglazov and Rajkumar Buyya. Adaptive Threshold-Based Approach for Energy-Efficient Consolidation of Virtual Machines in Cloud Data Centers. *Proceedings of the 8th International Workshop on Middleware for Grids, Clouds and e-Science*, (December 2010):6, 2011.
59. Anton Beloglazov, Rajkumar Buyya, Young Choon Lee, Albert Zomaya, et al. A taxonomy and survey of energy-efficient data centers and cloud computing systems. *Advances in computers*, 82(2):47–111, 2011.