The Journal of Systems & Software 175 (2021) 110907

Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss

FollowMe@LS: Electricity price and source aware resource management in geographically distributed heterogeneous datacenters[☆]

Hashim Ali [a], Muhammad Zakarya [a,*], Izaz Ur Rahman [a], Ayaz Ali Khan [a], Rajkumar Buyya [b]

[a] Department of Computer Science, Abdul Wali Khan University, Mardan, Pakistan
[b] Cloud Computing and Distributed Systems (CLOUDS) Lab, School of Computing and Information Systems, The University of Melbourne, Australia

ARTICLE INFO

Article history:
Received 30 September 2020
Received in revised form 9 January 2021
Accepted 12 January 2021
Available online 19 January 2021

Keywords:
Datacenters
Performance
Migrations
Energy efficiency
Clouds

ABSTRACT

With rapid availability of renewable energy sources and growing interest in their use in the datacenter industry presents opportunities for service providers to reduce their energy related costs, as well as, minimize the ecological impact of their infrastructure. However, renewables are largely intermittent and can, negatively affect users' applications and their performance, therefore, the profit of the service providers. Furthermore, services could be offered from those geographical locations where electricity is relatively cheaper than other locations; which may degrade the applications' performance and potentially increase users' costs. To ensure larger providers' profits and lower users' costs, certain non-interactive workloads could be either: moved and executed in geographical locations offering the lowest energy prices; or could be queued and delayed to execute later (in day or night time) when renewables, such as solar and wind energies, are at peak. However, these may have negative impacts on the energy consumption, workloads performance, and users' costs. Therefore, to ensure energy, performance and cost efficiencies, appropriate workload scheduling, placement, migration, and resource management techniques are required to mange the infrastructure resources, workloads, and energy sources. In this paper, we propose a workload placement and three different migration policies that maximize the providers' revenues, ensure the workload performance, reduce energy consumption, along with reducing ecological impacts and users' costs. Using real workload traces and electricity prices for several geographical locations and distributed, heterogeneous, datacenters, our experimental evaluation suggest that the proposed approaches could save significant amount of energy (∼15.26%), reduces service monetary costs (∼0.53% - ∼19.66%), improves (∼1.58%) or, at least, maintains the expected level of applications' performance, and increases providers' revenue along with environmental sustainability, against the well-known first fit (FF), best fit (BF) heuristic algorithms, and other closest rivals.

1. Introduction

Cloud computing offers utility-based services to IT users across the world. Its impact is increased with the demand of computing resources like CPU, storage access, network and applications for business, consumer or scientific domain. The hosting of these resources are provided by large datacenters. These datacenters, in return, consume large amount of energy, yielding a high cost for operation of these datacenters along with environment being affected by carbon footprints and green house gases (GHG) emitted[1] - i.e. *work most closely and frequently with carbon-free energy sources like solar and wind*. This is achieved through the idea of somehow *shift the timing of many compute tasks (non-urgent) to when low-carbon energy sources, like solar and wind, are most plentiful*. In 2016, it was projected that the world's datacenters utilized more than Britain's total power utilization — 416.2 Terawatt hours (TWh), essentially very higher than that of Britain's 300 TWh (Zakarya and Gillam, 2017a). As, accounting for approximately 3% of the worldwide power supply and approximately 2% of total GHGs, datacentres have almost same carbon emissions as of the aviation industry (Shehabi et al., 2016). The energy consumed by datacenters is approximately 205 TWh of electricity

☆ Editor: [J.C. Duenas].
* Corresponding author.
E-mail addresses: hashimali@awkum.edu.pk (H. Ali), mohd.zakarya@awkum.edu.pk (M. Zakarya), izaz@awkum.edu.pk (I.U. Rahman), ayazali@awkum.edu.pk (A.A. Khan), rbuyya@unimelb.edu.au (R. Buyya).

https://doi.org/10.1016/j.jss.2021.110907
0164-1212/

1 https://www.businessgreen.com/news/4014320/google-debuts-carbon-intelligent-computing-platform.

usage in 2018, which is nearly 1% of all electricity consumed worldwide, according to a new published report. The 205 TWh shows a 6% increase in overall power usage since 2010, though compute instances for global datacenter raised by 50% within the same period of time. This increase in energy consumption is due to the fact that there is an enormous increase in on-line services, mobile devices and users, on-line gaming, and IoT (internet of things) based devices.

Besides the above, Shehabi et al. (2016) also depicts that conceivably, due to migration of workloads, for the organization from private clouds to the public clouds, datacenters energy utilization will conceivably stay unaltered till 2020. These sorts of issues can be fathomed, mostly, through utilizing methods such as resource allocation, workload placement, scheduling, and consolidation i.e. efficient resource management techniques (Lebre et al., 2019; Verma et al., 2015). Resource management strategies are dependent over the already available technologies such as, virtualization and containerization — container-based virtualization. They are broadly utilized by cloud service providers to give resources to IaaS (Infrastructure as a Service) clients. Virtualization builds the idea of a VM (virtual machine) whereas containerization portrays the VM to as container; both running on virtualized servers. VMs have been broadly utilized in public clouds, especially, the state-of-the-art in IaaS is broadly aware with the notion of VMs. Cloud service providers such as Microsoft Azure, Google, and Amazon EC2 offer VM and container services to their clients and conjointly execute applications (workloads/services) inside VMs and/or containers. Besides, different PaaS (Platform as a Service) and SaaS (Software as a Service) suppliers, such as Google App, Gmail, are placed on top of IaaS where they execute all their applications and workloads inside VMs and/or containers.

The world is on track for perilous climate alter, having about misplaced room to assist contamination within the mix of gasses that make up the air. In spite of a rise in clean, renewable energy supplies in certain nations like the UK, Germany; and a fractional move from coal to natural gases in other countries, the worldwide GHG contamination still proceeds to rise — and at an expanding pace within the most later a long time (Zakarya and Gillam, 2019a). This alarms the need for energy-aware computation to be taken into account on a priority basis without any negative impact on applications' performance (Ferreto et al., 2011; Sharma et al., 2019). The renewable energy has reached up to approximately 54 TWh (3.3%) of the Britain's total energy utilization in 2010, having expanded consistently since 2005; and by approximately 15% from 2008 to 2009. We will expect, through these figures, more than a four times increment in the renewable energy utilization by 2020; in the event that approximately 15% of the energy requirements are to be met from renewable energy sources. The utilization of renewable energy will ought to rise by ∼17% annually to meet these objectives. A large proportion of datacenter usage, a main source of energy consumption, today is through the use of public clouds. Furthermore, it is estimated that in 2021, approximately 53% of worlds' all servers will be located in the hyper-scale public cloud datacenters. This basically means Amazon Web Services (AWS), Google compute cloud, Facebook, and Microsoft Azure (Shehabi et al., 2016).

The problem with contracting energy is that it is sort of cheating. Whilst, renewable energy is probably being generated somewhere, that may not be where your datacenter is located. A potential option to fix this issue is to deploy renewable sources of energy on the local grid providing power 24/7 a week; so that the datacenter can actually consume renewables at all times. This is much more difficult because of the varying locations of datacenters and unpredictable weather conditions (intermittent). Albeit, some IaaS facilities are located in regions with abundant renewables such as wind, solar and/or hydro while others are not. The Google team began work to achieve 24/7 a week available renewables in 2018. Furthermore, their approach towards carbon-intelligent computing[2] offers ways to shift workloads to times of day with peak renewable energy. It is drawing closer to the development of its claim, or contacting to third parties, sources for renewable energy that go specifically into the local network. Google published an article about their approach which incorporates a few interesting illustrations of the concept[2]. However, this is still not possible to switch all datacenter operations to renewable; because in 2018, approximately 63.5% of electricity generation in the United States was from fossil fuels such as coal (US Energy Information Administration, 2019). Furthermore, various regions offer different and varying prices for energy consumption. These will make the resource providers to run user workloads competitively for cheap energy sources and low prices to increase their money savings and ecological impacts. However, this should be optimized subject to network costs in terms of latencies and workload performance i.e. execution times (translating to user bills). This needs further exploration, investigation and research which is the focus of this paper (Liu et al., 2013).

In this paper, we investigate how workloads could be run in geographically distributed cloud datacenters so that the energy cost can be minimized without any negative performance impacts (Koronen et al., 2020). Moreover, how performance of workloads would be affected when putting or migrating them in locations with the least energy prices and higher availability of the renewable source. Google has taken initiative to shift workloads in their clusters according to time of the day; in order to increase environmental sustainability. However, the details of their approach are still not published. Moreover, to the best of our knowledge, with the notable exception of Xu and Buyya (2020), there is no study in current state-of-the-art datacenter approaches that considers migrating workloads across different clusters. Besides several limitations of our work, our findings are of interest and noteworthy with respect to energy savings, providers' revenue, and performance gains. Following are the major contributions of the research conducted in this paper:

- a placement policy "FillUp@LS" is suggested that puts appropriate workloads on appropriate clusters, according to energy sources and prices;
- a consolidation policy "FollowMe@Location" is proposed that migrates workloads across different clusters, geographically distributed, offering variations in energy prices, in an energy, performance, cost effective way;
- a consolidation policy "FollowMe@Source" is proposed that migrates workloads across different clusters fuelled through different energy sources, i.e. renewables, grid energy, etc., such that the workload performance is not affected, negatively;
- we investigate the energy, performance and costs' impacts of both "FollowMe@Location", "FollowMe@Source" policies; and how a combination of both these consolidation strategies "FollowMe@LS" would affect the infrastructure energy consumption, workload performance (execution times), and users' costs; and
- the proposed scheduler (placement plus consolidation) runs in a distributed fashion — where the global scheduler communicates with several local schedulers in order to take appropriate workload execution decisions.

The rest of the paper is organized as follows. In Section 2, we discuss the resource allocation, placement and consolidation problem in geographically distributed cloud datacenters along with

2 https://www.blog.google/outreach-initiatives/sustainability/100-percent-renewable-energy-second-year-row/.

variations in electricity prices and sources of production. In Section 3, we propose an allocation policy to put workloads on appropriate resources. Furthermore, we proposed two consolidation approaches i.e. "FollowMe@Location" and "FollowMe@Source" that prefer to migrate workloads among geographically distributed clusters according to electricity prices and sources of production, respectively. Both policies are, then, combined to come across a third consolidation method i.e. "FollowMe@LS" that take appropriate migration decisions to account for electricity prices and sources of production, simultaneously. We describe the simulation configuration, evaluation metrics, and different experimental parameters along with simulation models in Section 4. We evaluate and validate the proposed policies through real workload datasets from Google, in Section 5 and demonstrate its efficiency in terms of energy, performance and, therefore, cost with respect to existing methods. In addition, Section 5.5 briefly summarizes our experimental outcomes, validity of the obtained results along with limitations. In Section 6, we offer an overview of the related work. Finally, Section 7 summarizes the paper along with several shortcomings, limitations, and proposes future research directions.

2. Problem description

Largely, cloud service providers (CSP) use various sources, as shown in Fig. 1, to produce electricity that fuel their infrastructure, offices, cooling, and lighting devices etc. Furthermore, a single CSP may have different infrastructure or datacenters which are distributed over various geographical locations (e.g. the notion of availability zones in the Amazon web service cloud). Different energy sources and, as well as, geographical locations would have different prices for electricity at different times of the day.[3] Besides providing services at the edge level, CSPs would be interested to run user applications energy, ecological and cost effectively. For example, renewables are: (i) intermittent and may not be available any time; or (ii) renewables are cheaper than grid energy, as well as, environmental friendly. Therefore, certain workloads such as non-interactive (non-real time) tasks including YouTube video processing, could be run, as appropriate, to optimize these objectives. For example, when renewables (solar, wind) are at peak (time of the day), then, running workloads at maximum can be more effective. Moreover, certain workloads, non-interactive tasks, can be delayed for execution while taking benefits from renewables. Similarly, electricity prices varies from locations to locations, particularly in the United States, that could be of interest to CSPs in order to decrease their energy bills, therefore, increase their profits and/or reduce users' monetary costs. This could be achieved through VM placement, scheduling and consolidation with migration policies.

To face and solve these challenges, the design and implementation of an effective, and elastic scheduler and resource management approach to monitor the whole infrastructure is difficult, yet also essential. A scheduler is an integral and main part of a resource management system which is responsible to schedule jobs/VMs on appropriate resources. Usually, the scheduling problem is assumed as a bin-packing issue which is NP-hard; and is solved using numerous heuristic algorithms. Albeit, heuristics are not optimal, but they are enough fast to reach a scheduling decision. Other methods, such as backfilling (Tsafrir et al., 2007), are used to convert classical heuristics into approximate approaches. It is needed since collecting resource statistics makes it conceivable to yield proper adaptation decisions both at: (i) strategic level (e.g. the selection of one or more server where it will be executing at certain geographic region) and dynamic level (e.g. the resource reconfiguration, load-balancing, resource scaling, migration, re-allocation and so on). Such decisions should be taken in such a way that performance of the workloads is not negatively affected — since performance loss will translate to increased users' monetary costs. Furthermore, other essential objectives should also be guaranteed.

2.1. Problem formulation

The above problem can be assumed as a multi-objective optimization with focus to minimize energy bills, energy consumption (more ecological and environmental friendly as less energy consumption means low production), improve or, at least, maintain the expected level of performance, and reduce users' service costs. Note that, energy bill and energy consumption are directly proportional to each other and can be assumed as a single objective. Moreover, performance, when considered as workload execution time, is directly proportional to user' service cost (pay as you go) and can be assumed as a single objective. Furthermore, lower execution times mean improved performance and lower users; monetary costs — workload performance is an inverse of the workload execution time. Thus, the multi-objective problem is translated to an equivalent bi-objective optimization problem (Khan et al., 2020). The former objective can be denoted as $\mathcal{E}$ while the latter one as $\mathcal{C}$. Since, both objectives carry the same goals i.e. minimization; therefore, their product can be assumed as a single objective (Zakarya and Gillam, 2017a). Mathematically, the single objective of our bi-objective optimization problem can be written as:

$$minimize(\mathcal{E}.\mathcal{C}) \tag{1}$$

subject to several constraints such as: (i) the workload performance is not degraded; and (ii) each workload exactly runs at a single location or datacenter at a particular time. Besides, other constraints can also be available to form a multi-objective optimization problem (Xu et al., 2016). We solve the problem using the well-known heuristic techniques, such as First Fit (FF), Best Fit (BF) and so on. The total energy cost $\mathcal{E}$ is computed through multiplying the energy price E_{price} with the total energy consumption (EC) of n hosts in a particular cluster. Furthermore, the EC relate to real benchmarked values as described later in Section 4.2. The host and datacenter energy consumption is measured in kWh; while the energy price is measured in US dollars per kWh — thus the unit of $\mathcal{E}$ translates to dollars.

$$\mathcal{E} = E_{price} \times \sum_{host=1}^{n} EC_{host} \tag{2}$$

Similarly, the user monetary cost $\mathcal{C}$ is computed through multiplying the service price (VM_{cost}) (depending on the VM type) and execution time (or runtime) T of the workload. The workload execution time is the sum of all tasks' runtimes, which belong to a particular workload or application. The workload runtime is measured in hours; while the VM cost is measured in US dollars per hour — thus the unit of $\mathcal{C}$ translates to dollars.

$$\mathcal{C} = VM_{cost} \times \sum_{task=1}^{workload} Runtime_{task} \tag{3}$$

Note that, workload runtimes is inversely proportional to the workload performance i.e. lower performance values mean higher runtimes, there, higher users' monetary costs and vice versa. In certain circumstances, performance may be more preferably refer to response time e.g. real time cloud services (O'Loughlin, 2018); however, since users are billed based on their workload runtimes, therefore, we prefer this as a good performance metric (O'Loughlin, 2018).

[3] https://datacenterfrontier.com/google-shifting-server-workloads-to-use-more-renewable-energy/.

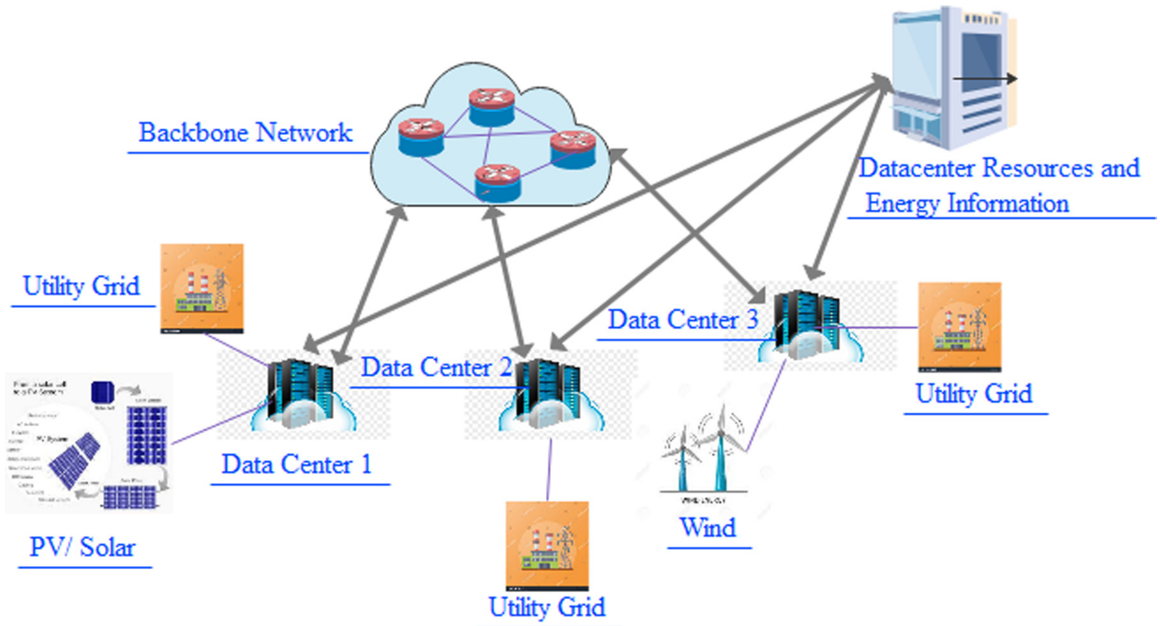

Fig. 1. Distributed datacenters with various energy sources and locations (Khosravi, 2017).

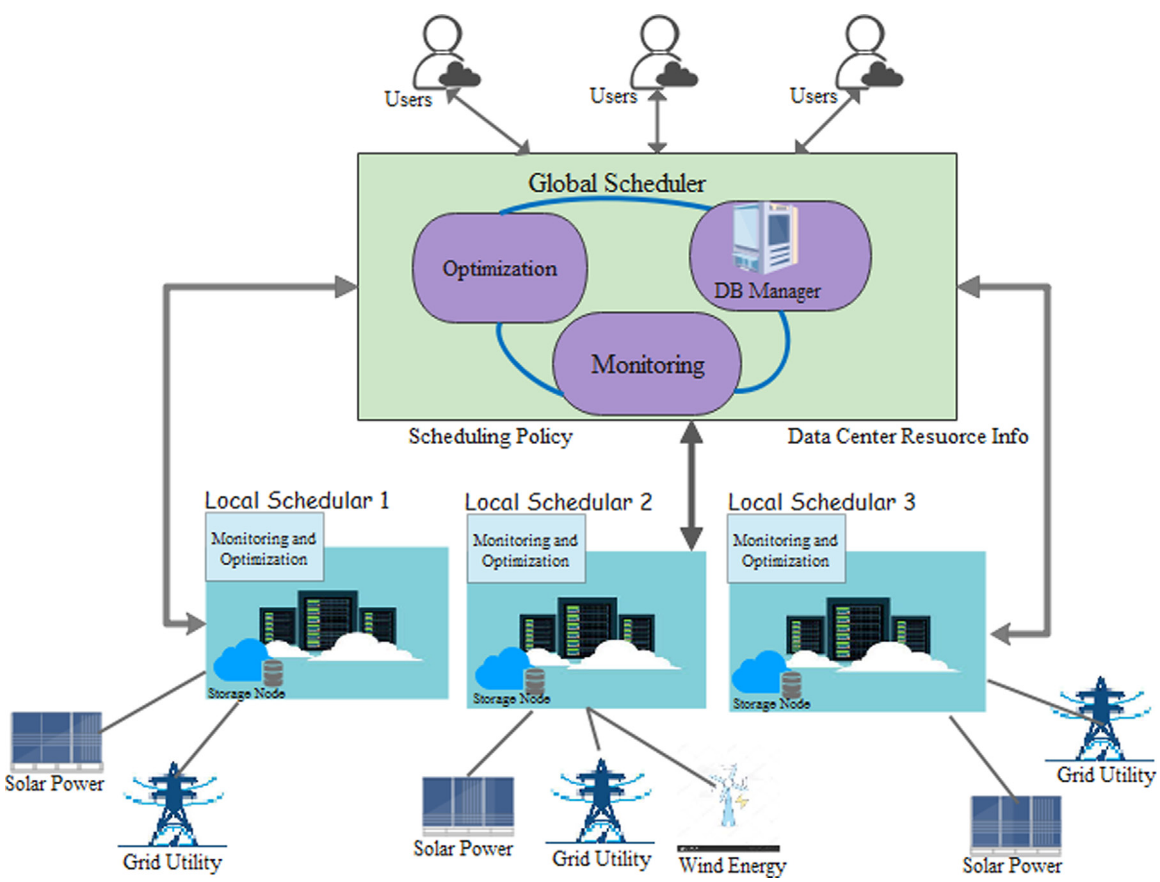

Fig. 2. Distributed scheduling across various datacenters.

3. Proposed FollowMe@LS technique

The above problem can be solved using heuristic techniques which are suggested to be more appropriate than optimal solutions, particularly, in large-scale online problems such as VM placement and consolidation (Tsafrir et al., 2007). Furthermore, VM placement can be assumed as sub-part of the consolidation with migration problem. During consolidation, a set of hosts (under-utilized and over-utilized) are considered. Then, a list of VMs are selected for migration from these hosts; Finally, the selected VMs are placed on appropriate hosts. This section describes a VM allocation and a consolidation policy in order to meet various objectives criteria. VM placement and consolidation decisions are usually triggered by the scheduler that might be a centralized or distributed, as shown in Fig. 2. A scheduler, or more specifically, cloud scheduler is an essential element in the cloud broker which is responsible to manage all infrastructure resources according to customers' requirements and quality of service (QoS). A single scheduler can be more appropriate as it will have the knowledge of all infrastructure but it suffers from single point of failure (Khan et al., 2019c). A distributed scheduler, at some additional cost of communication, could manage large number of heterogeneous resource more effectively.

In our proposed framework, each cluster (geographical datacenter) is in control of a particular local scheduler. The local scheduler is responsible to assign VMs to appropriate hosts in that particular cluster; and have a monitoring module to gather resource statistics such as utilization level. Each local scheduler has an optimization module that can take intra-cluster re-allocation decisions based the statistics which are gathered by the monitoring module and stored into a storage unit, preferably, a network area storage (NAS). On top of local schedulers, a global scheduler is responsible to take appropriate workload placement and intra-clusters migration decisions. Note that, the global scheduler uses data and statistics received from local schedulers in such affective decisions. Both schedulers use some kinds of VM placement and consolidation techniques in order to optimize various objectives. A single scheduler may take more appropriate workload allocation and migration decisions — as it has all statistics, knowledge and data of the clusters (Khan et al., 2019c); while a distributed scheduler involves additional costs of communication. Forthcoming sections describe the proposed policies for VM placement and consolidation.

3.1. VM placement policy

When a VM request is received, the proposed allocation strategy looks for a cluster that could run the VM on the lowest price based on energy source and/or electricity price for various locations. For example, if cluster A is fuelled through renewable while cluster B is using grid energy; then, cluster A is selected for the placement. Similarly, if electricity prices at location B' (location of cluster B) are lower than location A' (location of cluster A); then, cluster at location B' (i.e. cluster B) is preferred for allocation. To ensure further energy savings, most utilized hosts are allocated first; in order to guarantee that fewer hosts are in use. The process is repeated until an appropriate and economical host is allocated to the VM. In case, the VM request cannot be allocated due to non-availability of the required resources, it is added to the wait queue for scheduling in the next allocation round. The steps involved in the allocation process are described in Alg. 1. For implementational simplification, we can use the power usage effectiveness (PUE) as an evaluation metric to measure the energy efficiency of a particular cluster in relation to electricity sources. This means that a cluster with lower PUE than another cluster is using green energy source to power its infrastructure.

From step 1 to 4, clusters are settled up for appropriate allocation decisions through sorting them out on factors such as energy price and source. These steps can be modified as per the objective. For example, to account for both location and energy sources i.e. "FollowMe@LS", steps 1, 2, and 3 should be ignored; while to account for a single objective either step 2 ("FollowMe@Source") or step 3 ("FollowMe@Location") will remain while the others two steps i.e. 1, and 4 should be eliminated or commented from the pseudocode. Very similar to these approaches, the "FillUp@LS" VM placement policy can also be considered as: "FillUp@Source" which only accounts for energy sources; and "FillUp@Location" which only accounts for geographical locations offering cheap energy. Step 5 ensures that

Algorithm 1: FillUp@LS VM allocation policy

```
   Input: Clusters list (C), Hosts list (H), Wait queue (W), VMs
          list (V)
   Output: Price aware VM placement
1  sort C in increasing order of prices (locations and sources) ;
2  // for FollowMe@Source, sort C in increasing order of source
   prices ;
3  // for FollowMe@Location, sort C in increasing order of
   location prices ;
4  // for FillUp@LS and FollowMe@LS, sort C in increasing
   order of source × location ;
5  ∀ clusters ∈ C, sort H ∈ clusters with respect to available
   slots ;
6  for each vm ∈ V do
7      for each cluster c ∈ C do
8          for each h ∈ H do
9              if h is active and has enough resources to run the
               vm then
10                 allocate vm to h using an appropriate
                   placement policy;
11                 break the loop and pick the next vm ∈ V;
12             end if
13         end for
14     end for
15     if vm cannot be allocated to any active h ∈ H then
16         start a new h′ ∈ H and assign vm to h′;
17     else
18         "vm can not be allocated";
19         "add the vm request into W";
20     end if
21 end for
```

all hosts in different clusters are sorted based on the available capacity. This guarantees that few hosts are most utilized within the cluster to reduce energy consumption. From step 6 to 14, each VM is placed according to some sort of placement policy e.g. FF, BF, FillUp, etc. If a VM cannot be placed on a switched on hosts (h), then a new host (h') is switched on and the VM is placed there (step 15 to 17). Unfortunately, if there is no appropriate host (step 18 to 21), then the VM is paced on to a waiting list (W). VMs in the waiting list are rescheduled periodically.

3.2. VM consolidation policy

VM consolidation can be achieved through migration (Zakarya and Gillam, 2019a). During a migration, a VM is moved from one host to another host. If the VM is transparently being moved while the service inside the VM is running during the migration duration, then the migration is called live (Ferreto et al., 2011). Usually, migrations are used to decrease the energy consumption of datacenter resources through consolidating the workloads on fewer hosts while switching off or turning unnecessary hosts into low power consumption mode. We, here, assume migrations for the purpose of reducing energy consumption, as well as, utilizing lower energy prices, if available. In the first round, we identify all clusters locations and energy sources that fuel them. In the second phase, all under-utilized and over-utilized hosts are marked, VMs are collected for migration, and a particular allocation policy is used to place them where appropriate. The steps involved in the consolidation process are described in Alg. 2.

From step 1 to 5, appropriate clusters are identified based on the energy price and source (PUE). Further, details of all hosts are

Algorithm 2: FollowMe@LS VM consolidation policy

```
Input: Clusters list (C), Hosts list (H), Energy sources (S),
       Prices (P)
Output: Price aware VM consolidation
1  foreach cluster do
2      identify cluster energy sources S [ignore this step to
       implement "FollowMe@Location"];
3      identify cluster location and electricity prices P [ignore
       this step for "FollowMe@Source"];
4      gather cluster info, hosts statistics and workloads
       details;
5  end foreach
6  foreach consolidation round do
7      platform.optimize(C, H) ;
8      L ← all VMs that need to be migrated ;
9      foreach cluster do
10         for h in H do
11             if h is under-utilised | over-utilised then
12                 mark appropriate VMs on h in cluster ;
13                 add marked VMs to L ;
14             end if
15         end for
16     end foreach
17 end foreach
18 for vm ∈ L do
19     // abort migration if target host is not within the same
       cluster (for intra-cluster migrations) ;
20     if targetHost.ClusterID ≠ sourceHost.ClusterID then
21         abort migration [for intra-cluster migrations];
22     end if
23     allocate vm using Alg. 1 ;
24     indicate allocation option i.e. source prices, location
       prices or both ;
25 end for
```

gathered from the storage nodes. From step 6 to 17, these steps are repeatedly run in each consolidation round (periodically). These steps look for migration opportunities and all migratable VMs are placed in a list. From step 18 to 25, all migratable VMs are scheduled for placement using Alg. 1. Note that, Alg. 1 should be modified according to the desirable heuristic approach such first fit (FF), best fit (BF), and FillUp (Zakarya and Gillam, 2017a). In a similar way, Alg. 2 should also be modified as per the desirable migration policy. For example, step 2 must be ignored for the implementation of the "FollowMe@Location" policy while step 3 must be eliminated for the "FollowMe@Source" policy. This is due to the fact that the "FollowMe@LS" policy accounts for both: (i) variations in prices due to geographical locations; and (ii) various energy sources like coal, renewables. The "FollowMe@Location" migration policy prefers to migrate VMs to locations having the least energy costs. Furthermore, the "FollowMe@Source" migration policy prioritize migrations to clusters operated for cheaper and renewable energy sources. From implementation point of view, this could be easily achieved through sorting (as appropriate) the available hosts, having enough capacity to accommodate the migratable VMs, based on their locations and/or energy sources. Approximate and optimal algorithm can also be added to the description of Alg. 1, for more affective, energy, performance and cost-efficient VM allocation.

The above consolidation approach can migrate VMs either: (i) inside a particular cluster (among different hosts); or (ii) across several clusters (among various hosts which may belong to different clusters). The former one is known as *intra-cluster* migration while the latter one is called *inter-clusters* migration. From implementation point of view, intra-cluster migrations can be assumed as a migration control policy – a check over the target host in Alg. 1; if the target host is not within the same cluster as the source host, then, the migration can be aborted and the next migratable entity is selected from the list of all migratable VMs (Khan et al., 2019b). However, reducing the number of migrations could leave the cluster resources more stranded; therefore, may lead to lower energy efficiency. There are various ways to reduce the number of stranded resources e.g. backfilling; which allows allocating VMs/jobs with certain characteristics to fill the gaps. More formally, the scheduling heuristic can be optimized to account for these gaps through sorting out the list of VMs in a particular order. The former one is most suitable for on-line problems while the latter one is most appropriate for off-line problems. In Section 6, we describe various scheduling heuristic approaches. The proposed allocation and migration policies can be easily modified to account for a particular choice. In our evaluation, we perform both kinds simultaneously; as putting a constraint over the migration can reduce migration opportunities and, therefore, is less economical. Increased number of migrations may mean lower performance because a VM migration can reduce the running workload performance approximately 10% (Khan et al., 2019b). Besides performance degradation, migrations also consume additional energy because two VMs are running for the duration of the migration (Zakarya and Gillam, 2016). In our evaluation, we account for migration energy and performance costs both. Moreover, we also account for performance variations (in applications' runtimes) which may happen due to CPU architectural design, heterogeneity, co-location, and resource interference, as described later in Section 4.3. In this paper, the following three variants of the proposed VM consolidation policy are considered for the performance evaluation of various workload using empirical experiments.

FollowMe@Location: The "FollowMe@Location" puts or migrates workloads across different clusters, geographically distributed, offering variations in energy prices, in an energy, performance, cost (EPC) effective way. This policy can be used to run the delayed workloads and (non-interactive) cloud services at later times when prices in certain locations drops, dynamically.

FollowMe@Source: The "FollowMe@Source" puts or migrates workloads across different clusters fuelled through different energy sources, i.e. renewables, wind, gas, grid energy, etc., such that the workload performance is not affected, negatively. This policy is affective if there are cluster IaaS resources that are powered using renewables; that could be intermittent.

FollowMe@LS: The "FollowMe@LS" combines the above two policies in order to account for geographical location prices (dynamically changes with respect to time, demand, and usage) and energy sources that run various geographically distributed clusters. From VM placement point of view, we account for both i.e. "FillUp@LS"; however, when migrating workloads then all three variants are considered in this paper. However, the "FillUp@LS" policy can be easily modified to account for just a single objective i.e. location, price.

3.3. Implementation methodology

Despite the large volume of research available on VM consolidation with migrations, there are only few software tools available online that support consolidation and are used to design geographically distributed clouds. In the literature, the earliest open-source implementation of server consolidation is Entropy.[4] A second framework for VM management in private

4 http://entropy.gforge.inria.fr/.

clouds called Snooze.[5] A third open-source implementation of OpenStackNeat,[6] a framework for server consolidation in OpenStack clouds. An overview of these consolidation systems can be found in our previous studies (Zakarya and Gillam, 2017a; Khan et al., 2019a). We believe that a discussion of such tools and implementation will help our readers to understand how the proposed resource management techniques (allocation, consolidation through migrations) would be implemented in real production cloud environments. These platform can be designed in a private IaaS cloud that can be easily updated regarding different VM placement, consolidation, and resource management policies. Unfortunately, it is very difficult to conduct experiments in real public cloud, as the policies are not directly accessible.

The main requirement for the implementation of the proposed algorithms is that a full and functional real test-bed, which runs a hypervisor along with any cloud management tool [such as Entropy, Snooze, OpenStackNeat], is available. Furthermore, for these systems, the global manager must be installed on a particular server that runs on top of several local managers which are running over servers connecting different clusters. Then, for each cluster the local manager does the same job of a global manager connecting various servers (Tchana et al., 2016). The consolidation technique might be implemented in a distributed fashion by running the consolidator part (i.e. VM selection algorithms) on every compute host and the other part (i.e. VM allocation algorithm) on a separate controller host. The core of the OpenStack lies in the compute module (Nova), which is responsible for VMs provisioning and management. During VMs provisioning, Nova uses Glance that is a repository for instance types. The Nova scheduler is responsible for VMs placement onto hosts that, by default, uses either: (i) the chance/random mechanism; or (ii) the filter & weight approach. This scheduling approach can be easily replaced with the proposed policy — weight the available hosts with respect to slots available (utilization), their energy consumption and performance. Nova compute provides key metrics such as: (i) hypervisor-based metrics *[hypervisor_load, current_workload, running_vms, vcpus_available]*; (ii) tenant-based metrics *[total_cores_used, total_instances_used]*; and (iii) Nova server-based metrics *[hdd_read_req]*; that can be useful to determine resource utilization, energy consumption and performance.[7] External monitoring tools such as Zabbix,[8] Ganglia[9] and DataDog[10] can also be used to get usage data at specific intervals (e.g. 5 min) that the scheduler can use in VM placement decisions.

In a virtual platform, the hypervisor, that has access to all VMs, is responsible to consolidate the workload (VMs) when needed. Nova and docker support both cold (off-line) and live migration of VMs; and the migration approach can be located in the Nova manager API. In order to implement the proposed approaches, the code needs to be modified in two ways: (i) migrations can be triggered automatically each after 5 min intervals; and (ii) the data collected by the monitoring API can be used by the scheduler to place migrated VMs to destination hosts. Beloglazov and Buyya (2015) proposed a framework based on the OpenStack project that is able to initiate VM migrations (global manager — controller node) based on the host utilization thresholds (local manager — compute node). The proposed framework has data collector APIs that are responsible to send compute nodes statistics to the global manager for VM migration and placement decisions. Another framework for software consolidation, which closely resembles our proposed framework, has been suggested in Tchana et al. (2016); where each host and the hosts with local managers has a monitoring agent that gathers local statistics and send them to the monitoring engine. The consolidation manager runs periodically, on a separate host along with the monitoring engine; collects data from the monitoring engine in order to decide reconfiguration plans (migrations) and informs the local manager (on each host) to take appropriate action. The price and renewable models of our framework can be considered as part of the consolidation manager; while energy and performance data is collected on every host and stored on a shared storage.

4. Simulation configuration

We modelled and simulated geographically distributed clouds in order to evaluate the performance of the proposed allocation "FillUp@LS" and consolidation "FollowMe@LS" policies. To ensure accuracy, plausible simulations were based on plausible and realistic models and real workload traces. CloudSim (Calheiros et al., 2011) is one of the most widely used simulators in the cloud research community, which offer an easy way to model distributed clusters. We consider real workload traces from the Google cluster (Reiss et al., 2011) and Microsoft Azure (Cortez et al., 2017). The former one is captured in a containerized platform[11] while the latter one comprises records of VM instances.[12] In both datasets, each task (assumed as running part of a particular workload in a container or a VM) has certain characteristics like arrival or submission time, resource (CPU, memory, disk) demand and actual usage, submitting user, and finish time. Moreover, both datasets include seasonal aspects, burstiness, and other important features that can be of interest. For our study, VM arrival times (arrival rate), resource usage, and execution times are very important. The execution time or runtime of each task is computed through subtracting its submission time from the finish time. Note that, in our simulations the arrival time of each VM exactly matches the arrival of tasks in these datasets. Moreover, as users pay for their resources based on their capacities and usage time (PAYG — pay as you go model); thus, we believe that VM execution time can be a good performance metric for certain types of cloud workloads or applications.

4.1. Evaluation metrics

We consider total number of migrations (intra-cluster and inter-clusters), energy consumption (kWh), workload performance (execution time measured in minutes or hours), energy bills (in dollars), and user' service costs (in dollars) as the performance evaluation metrics. The intra-cluster migrations are those which may occur among various hosts of a single, particular, cluster; while inter-clusters migrations may happen among hosts which belong to different clusters/datacenters. The total energy consumption of each host is the sum of energy consumption of all VMs accommodated on that particular host. The energy consumption of each VM is computed, using Eq. (4), which is a fraction of the host's benchmarked energy consumption values. Moreover, the workload performance is the sum of all VMs execution times that run the workload. Similarly, the energy bill refers to the amount (in US dollars) of the total energy used; which is computed dynamically based on the geographical location, time, and energy prices (real benchmarked values in the US)[13] – as shown in Table 1. Finally, the service cost is the sum of amount (in US dollars) of all VMs that run the given workload of a particular

5 http://snooze.inria.fr/.
6 http://openstack-neat.org/.
7 https://www.datadoghq.com/blog/openstack-monitoring-nova/.
8 https://www.zabbix.com/zabbix_agent.
9 http://ganglia.info/.
10 https://www.datadoghq.com/.
11 https://github.com/google/cluster-data.
12 https://github.com/Azure/AzurePublicDataset.

Table 1
Datacenters geographical locations, energy sources and prices.

Datacenter site	Energy price (cents/kWh)	Energy source (PUE)
DC1-Richmond Virginia	6.54	1.9
DC2-San Jose California	10	1.7
DC3-Portland Oregon	5.77	1.56
DC4-Dallas Texas	6.1	2.1

user on PAYG model — computed as shown in Table 3. We are aware that there would other performance metrics; however, we believe, execution time as a good performance metric; as cloud users pay for their services based on their provisioned resources, their capacities, and usage times (O'Loughlin and Gillam, 2014).

4.2. Experimental set-up

In order to evaluate the proposed algorithms, we modelled a geographically distributed IaaS cloud with 4 datacenter sites, as shown in Table 1, and each cluster in different geographical location have 2500, 3000, 4000, and 5000 heterogeneous hosts, respectively. All datacenters are interconnected to each other using same network bandwidth i.e. 1 Gbps, but, having different network distances i.e. communication costs. These costs matter, in particular, during moving workloads among different clusters. We assume that the global scheduler is aware of all these distances. Each datacenter has a unique PUE. The PUE values refer to the work presented in Khosravi et al. (2017) which represent the energy efficiency (source) of a particular cluster in a specific geographic area. Note that, PUE is used here as notion to represent the source of energy which, in practice, is not essential. These hosts relate to 7 different architecture types (CPU models) and shown in Table 2. Furthermore, we assume electricity prices at certain locations that reflect the real market prices in the United States.[13] In reality, prices vary with respect to time of the day. However, for implementational simplification, we assume that these prices remains unchanged. The energy consumption of these hosts relate to real benchmarked values at various utilization levels from the SPECpower.[14] In order to account for peak demand and burstiness of the workloads, the arrival time and inter-arrival ratio of VMs exactly match the submission times and inter-arrival rate of tasks in both real datasets; with the only exception of VMs wait times in the queue. This means that the placement policy deals with unknown workloads (VMs); however, the consolidation policy runs over known VMs (reserved) to pack them onto available servers in a more appropriate way.

The speed of each host is, then, mapped to millions of instructions per second (MIPS) in order to be consistent with the simulation platform i.e. CloudSim (Calheiros et al., 2011). Each host is modelled as virtualized which has the capability to run several VMs subject to the host's capacity — also known as the notion of VM density somewhere else (Zakarya, 2018b). The performance parameters for these various hosts running different applications (benchmarked over real IaaS experiments) are shown in Table 4. We assume different sizes of VMs (instance types) as shown in Table 3 – that reflect Amazon Web Services (AWS) instance types; while their performance, in terms of execution times, on various hosts are shown in Table 4. Each VM costs a particular user depending on the resource capacity and geographical location (Zakarya and Gillam, 2017a). To evaluate the performance of the proposed policies under this plausible simulation environment, we use real cluster data traces from Microsoft Azure cloud (Cortez et al., 2017) and Google (Reiss et al., 2012). For more realistic scenarios, these workloads were mapped to different applications, using statistical methods, as described later in Section 4.3. The former one is logged in a virtualized platform while the latter one is logged in a containerized platform. Furthermore, the former one consists of tasks with longer runtimes; while the latter one consists of task with short durations. Each workload consists of more than a million tasks and each task has certain characteristics such as runtime, schedule time, resource requirements etc. We further assume, that all tasks uses their CPU resources using a built-in CloudSim model i.e. stochastic utilization approach (Beloglazov and Buyya, 2012). As a whole, we assume all tasks in a workload as a single application whose execution time is the sum of all tasks' execution times. Further, we assume that each VM can run at most one task at a time. These assumptions allow us to map application execution in a cloud environment, and we believe that these are sensible ways to carry out these in a simulated platform.

In order to optimize the states of various clusters and minimize the energy consumption, we assume that VMs are being migrated: (i) inside a cluster (among hosts within a single, particular, cluster); and (ii) across several clusters (among hosts that may belong to different clusters). Such events occur when the resource utilization levels of hosts increases or decreases some pre-defined threshold values. The former case avoids performance degradation due to resource over-subscription and the later one can switch off under-utilized hosts to save energy. We assume that over-subscription does not happen due to ways and constraints over VM placement i.e. a VM cannot be placed on a host which does not have enough capacity to run it. For the later one, we set a threshold of 20% i.e. if utilization level of a host decreases than 20%; then, all accommodated VMs or workloads on this host are migrated to some other host. Moreover, if there are rooms to run VMs on cheaper energy on a particular cluster; then, appropriate VMs from other clusters are being migrated here. The optimization module runs periodically each after 5 min intervals and looks for migration opportunities. Furthermore, other approaches, such as on-demand, can also be used to trigger migrations and optimize the states of the datacenters. Very frequent runs of the optimization modules may significantly affect the total number of migrations, therefore, applications' performance, and infrastructure energy consumption.

4.3. Statistical models

This section describes how energy consumption of hosts/VMs and performance of hosts/VMs are modelled for simulation purposes. Furthermore, we also discuss how migration happens and its impact on energy consumption and workload performance degradation. These models are selected in such a way that a plausible and realistic simulation platform can be developed to ensure accuracy of the obtained results and outcomes.

Energy consumption: We use real benchmarked values from SPECpower[15] for the energy consumption of various servers, as shown in Table 2. However, the energy consumption of each VM is computed, using the linear power model, as given by Eq. (4):

$$P_{VM} = \left(\frac{P_{idle}}{N}\right) + W_{VM} \times (P_{peak} - P_{idle}) \times U \tag{4}$$

where P_{idle} and P_{peak} denote the energy consumption of a particular server when it is 0% and 100% utilized, respectively. Note that, the server energy consumption values were taken from the SPECpower benchmarks that are noted at various utilization levels i.e. 0%, 10%, 20%, 30%, and so on. Moreover, W_{VM} is the

13 https://www.eia.gov/electricity/.
14 www.spec.org.
15 https://www.spec.org/power_ssj2008/.

Table 2
Various characteristics of hosts for Amazon's cloud (simulated).

CPU model	Speed (MHz)	No of cores	No of ECUs	Memory (GB)	P_{idle} (Wh)	P_{max} (Wh)	Amount			
							DC1	DC2	DC3	DC4
E5-2630	2300	12	27.6	128	99.6	325				
E5430	2830	8	22.4	16	166	265				
E5-2620	2000	12	24	32	70	300				
E5645	2400	12	28.8	16	63.1	200	2500	3000	4000	5000
E5-2650	2000	16	32	24	52.9	215				
E5-2670	2600	16	41.6	24	54.1	243				
E5540	2500	4	10	72	151	312				

Table 3
Amazon different instance types and their characteristics.

Instance type	No of vCPUs	No of ECUs	Speed (MHz) MIPS	Memory (GB)	Storage (GB)	Reserved price (1 year) ($/h) 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

fraction of host resources allocated to a particular VM e.g. number of cores or vCPUs; and N refers to total number of VMs on a particular host. This model equally divide the idle power consumption of a host among various VMs running on it; however, more fair and approximate division may be possible (Khan et al., 2020). The energy consumption of a VM migration is computed according to the model presented in Liu et al. (2011) - energy use is proportional to the amount of VM memory (VM_{data}) to be moved from source to destination server; and is given by Eq. (5):

$$E_{mig} = 0.512 \times VM_{data} + 20.165 \quad (5)$$

The above model is validated for intra-cluster migrations; however, inter-clusters migration will usually take longer depending on the network conditions. For the latter case, we compute the migration time through dividing the VM data by the network bandwidth (assuming constant). Later on, the time is translated to energy consumption of the network plus source and destination hosts. To simulate VM migration across several clusters (migration time and downtime), we integrated the migration model used in *VmigSim* simulator[16] in CloudSim (Calheiros et al., 2011). The *VmigSim* simulator offers a realistic environment to mimic on-line VM migration (pre-copy) in different rounds; using various parameters for network, VM memory, page dirty rate, etc. Further details on the migration modelling, approaches, calculating durations and downtimes can be found in our previous works (Zakarya and Gillam, 2019a; Khan et al., 2020).

Performance: As investigated in O'Loughlin (2018), workload performance vary with respect to CPU platform i.e. similar workloads (applications) will run differently on same or different VMs (instance classes) accommodated on servers having different CPU architectures, as shown in Table 4. The mean (μ), standard deviation (σ), minimum (Min), and maximum (Max) runtimes (performance) of three different applications are shown in Table 4. The coefficient of variance (CoV) is computed through dividing σ over the μ; the smaller ones denote lesser variations in runtimes. The benchmarked values denote a log-normal distribution (O'Loughlin, 2018). Therefore, to represent CPU heterogeneity and host performance, we also assume that workload runtimes on different hosts are log-normally distributed. From implementation point of view, when VMs are being migrated from one host to another; the increase or decrease in runtime is computed from a log-normal distribution dataset. The process comprises translating the remaining execution time of a particular application running on a server (source) to equivalent execution time on a destination server. This could be done through the standard score (or more formally the z-score normalization method) (Zakarya and Gillam, 2019a). The standard score, as given by Eq. (6), is normally used to calculate the probability or likelihood of a particular score (r) which occurs in the interior of different datasets (normally distributed), given its statistics like mean (μ) and standard deviation (σ). Furthermore, z-score also provides a way to relate more than one scores with may or may not belong to various datasets which are, essentially, normally distributed.

$$z_{score} = \frac{r - \mu}{\sigma} \quad (6)$$

Eq. (7) could be utilized to compute the expected execution time of a migrated application (workload), from the source server, on the destination server given their distributions (usually normally distributed) along with their statistical means (μ, μ') and standard deviations (σ, σ') of source and destination servers, respectively.

$$\frac{r - \mu}{\sigma} = \frac{r' - \mu'}{\sigma'} \quad (7)$$

Note that, both r and r' denote the estimated runtimes of the migrated application on the source and destination servers, respectively. Furthermore, the left hand and right-hand sides of Eq. (7) narrate to the standard scores of the source and destination servers, respectively. The above mathematics allows us to predict the probable scores i.e. VM runtimes (translating to the expected increase or decrease in applications' performance on the destination server) occurring within a dataset which is essentially normally distributed. Note that, the dataset consists of the performance (runtime) dissimilarities due to resource, workload and/or platform heterogeneities, as shown in Table 4. The above Eq. (7) can be rewritten as Eq. (8), in order to compute the expected execution time (or workload performance - r') of the migrated application on the destination server given its estimated remaining execution time (r) on the source host:

$$r' = \sigma' \times \left\{ \frac{r - \mu}{\sigma} \right\} + \mu' \quad (8)$$

16 http://www.github.com/.

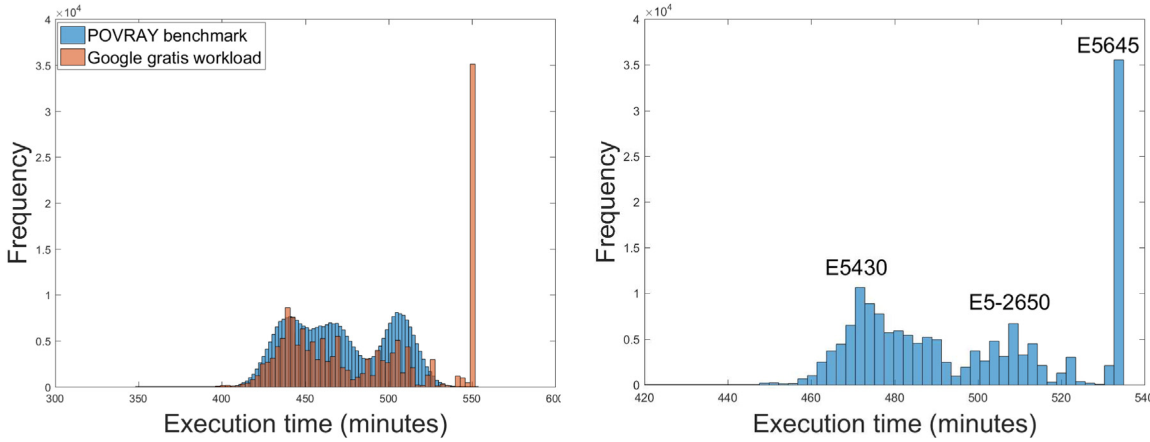

Fig. 3. Mapping the Google data to real benchmarks (left) and plausible assumptions for choosing appropriate hosts (right) (Zakarya and Gillam, 2019a) [Povray workload performs best on E5430 and worst on E5645].

For log-normally distributed datasets, both r and r' should be replaced with $log(r)$, $log(r')$ subject to the mathematical definitions of both normal and log-normal distributions (Zakarya and Gillam, 2019a). For log-normally distributed datasets, the estimated execution time (r') can be calculated using Eq. (9):

$$r' = exp\left(\sigma' \times \left\{\frac{log(r) - \mu}{\sigma}\right\} + \mu'\right) \tag{9}$$

We are aware that there would be more effective ways to estimate and predict the estimated or remaining runtimes of applications. Moreover, the overlaps which may exist in the performance of multiple servers for similar applications can also be accounted for. Methods like euclidean distance can be used to face and deal with similar overlaps. However, we keep it simple and, thus, assume no overlaps. These overlaps can be assumed as redundant data and methods like the euclidean distance can possibly remove these overlaps. However, this needs further investigation in order to associate these redundant data points (runtimes) with an appropriate CPU model instead of ignoring it at all. For example, under what conditions/parameters a particular application will essentially perform the same on two or more than two different servers and vice versa. Application performance is very important, in particular, when users pay for their resources using a PAYG model. As, application runtimes play a major role in cloud business economics (users pay for resource usage based on time); thus, we believe that execution time can be a good and more appropriate performance measurement unit to IaaS providers. The above statistical model is used in our previous works (Zakarya and Gillam, 2017a; Khan et al., 2019a) to account for increase or decrease in the execution time of a migrated application on two dissimilar CPU platforms. The means and standard deviations of applications' execution times are taken from real benchmarked values, as shown in Table 4.

Furthermore, we assume a 10% performance degradation in the workload of each VM which is being migrated intra-cluster (Beloglazov and Buyya, 2012). From implementation point of view, when a VM is migrated from one server to another then its remaining execution time is increased with 10%. For inter-clusters migration, an existing model (implemented in VmigSim simulator) is used to compute the downtime which is, then, added to the remaining runtime of the VM on the destination server. However, to account for performance loss due to CPU heterogeneities, we use the concept of z-score normalization and the law of log-normal distribution, as described in our previous works (Zakarya and Gillam, 2019a; Khan et al., 2020). For intra-cluster migration, the downtime of each migrated VM is translated to an equivalent degradation, as discussed in the above section. Besides these, VMs competing for similar resources (when running same workloads) while co-located on a single host may also suffer from severe performance degradation (O'Loughlin and Gillam, 2016). However, to make it simple, we do not account for these costs in our current work. Findings in O'Loughlin (2018) ascertain that performance can be severely degraded which may be as high as 42%. Furthermore, when resources are over-subscribed or overload; then VMs may suffer from performance issues. However, we assume no over-subscription; and the notion of VM density i.e. each host can accommodate number of VMs with sum of capacities less than the host entire capacity (Zakarya and Gillam, 2016), which does not result in over-load situations. In fact, our allocation policy check the available capacities of a host before assigning them to a VM. In case, resources are not enough, then the allocation is rejected.

Applications: In order to make our simulations more realistic, we mapped the Google (Reiss et al., 2011) and Microsoft Azure (Cortez et al., 2017) workloads to certain benchmarked applications in a real IaaS cloud. To do so, we used the performance values (means, standard deviations), as presented in O'Loughlin (2018), of three real applications i.e. Bzip2, Stream, and Povray. The authors suggest that performance of various instances in AWS EC2 cloud on similar CPU architecture significantly varies and can be modelled as log-normally distributed. Furthermore, combination of various CPU models and instance types produces multimodal distributions. Since, the offered data is not enough to get findings; therefore, we ran Monte-Carlo simulations to produce more data using the laws of normal and log-normal distributions. In next steps, the actual benchmarked values (i.e. application runtimes) and tasks' runtimes in both workloads i.e. Google and Microsoft Azure, were put into appropriate multimodal distributions (log-normal). Finally, close similarities in the distributions were assumed as tasks which may belong to certain real benchmarked applications on different CPU platforms (Zakarya and Gillam, 2019a). For example, Fig. 3 (left-hand side) demonstrates the actual benchmarked runtimes of the Povray application and the tasks' runtimes which belong to certain records in the Google dataset. After normalizing these values, the Google tasks' runtimes were mapped to the performance of the Povray application on appropriate hosts, as shown in Fig. 3 (right-hand side). Note that, Table 4 describes the performance parameters of three applications over different CPU platforms. Similarly, Fig. 4 mimics the performance of the Stream application, mapped to Microsoft Azure cloud dataset, over different CPU architectures. Various distributions were converted onto the same scale and, then, mapped

Table 4
Various applications' runtimes over different CPU architectures and instance types [696 MB input file to Bzip2 - Ubuntu 10.04 AMD desktop ISO file] (Zakarya and Gillam, 2019a; O'Loughlin, 2018; O'Loughlin and Gillam, 2014) — for example, Bip2 on *m1.small* VM type performs better when hosted on E5507 while Povray performs worse on the same CPU; while on *m1.medium* VM type, both applications' performance is the other way around (opposite); O'Loughlin (2018) suggests that these variations can be mapped to log-normal distribution.

Application type	CPU model	*m1.small*					*m1.medium*				
		(μ)	(σ)	Min	Max	CoV	(μ)	(σ)	Min	Max	CoV
Bzip2	E5430	445.1	14.33	425.35	482.1	0.032	211.30	10.43	204.71	238.2	0.049
	E5-2650	470.24	13.03	443.48	518.92	0.028	223.40	3.84	217.81	233.51	0.017
	E5-2665	241.3	1.18	237.97	245.2	0.005	–	–	–	–	–
	E5645	510.07	10.51	487.95	543.8	0.021	244.71	2.90	240.9	254.11	0.012
	E5507	620.87	28.46	578.03	715.72	0.046	312.92	14.91	295.91	332.01	0.048
Stream	E5430	693	3.0	687	701	0.004	196.21	15.03	174.56	245.86	0.077
	E5-2650	614	5.0	606	624	0.008	224.34	8.34	215.58	235.67	0.037
	E5645	606	7.0	599	628	0.012	230.83	12.19	216.93	250.03	0.053
	E5-2665	59.2	1.88	52.16	65.0	0.032	–	–	–	–	–
Povray	E5507	632	5.0	625	650	0.008	261.63	18.84	241.46	356.92	0.072
	E5540	623.9	3.2	612.5	636.8	0.005	241.1	2.9	231.9	250.7	0.012
	E5-2630	128	2.0	120.5	134.2	0.016	–	–	–	–	–
	X5560	525.5	0.6	524.4	526.8	0.001	–	–	–	–	–

Table 5
Execution times (seconds) of various applications on co-located VMs (Xu et al., 2016).

Workload type	CPU model	Number of co-located VMs					
		2	4	6	8	10	12
		Execution times					
Grep	E5620	13	14	16	21	31	36
	E7420	20	22	25	29	38	44
Sort	E5620	16	22	38	59	69	78
	E7420	21	28	43	65	76	85

using a simple visualization methods through identifying the total number of peaks, multi-modals, and thus CPU platforms and architectures (Zakarya and Gillam, 2019a).

Apart from the above discussion, co-located VMs on a particular server may experience severe performance loss, specifically, if the hosted virtualized applications compete for similar resources (also known as resource interference). Xu et al. (2016) empirically evaluated and observed that the performance loss is strongly dependent on the total number of co-located VMs and the application type they are executing on a particular server i.e. the more number of VMs co-located on the server, the worse will be its performance and vice versa — as shown in Table 5. It is also essential to account for such resource interference and contention costs among different servers. From an implementation point of view and in order to model performance variations of various applications on different CPU platforms, we model: (a) resource interference as a simple regression line equation with respect to total number of co-located VMs on a particular server for certain types of applications — based on prior studies in Xu et al. (2016, 2014); and (b) CPU platform heterogeneity as log-normally distributed with respect to application runtimes based on prior research and findings (Zakarya and Gillam, 2019a; O'Loughlin, 2018). The above mathematical models were used to design a sensible, and realistic simulation environment i.e. PerficientCloudSim (Zakarya et al., 2020) which is available on the GitHub repository.[17] Moreover, datasets were mapped to cloud applications using plausible assumptions. Similarly, resource and application heterogeneities, in terms of performance degradation, were modelled using sensible ways to certain benchmarked results as demonstrated in prior studies (O'Loughlin, 2018).

[17] https://github.com/mohd-zakarya/PerficientCloudSim.

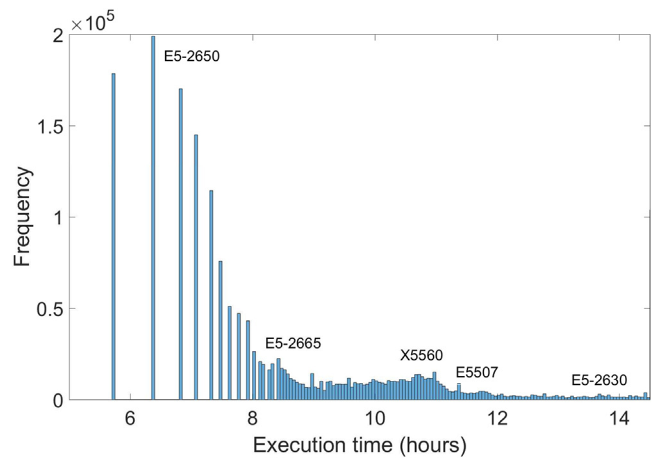

Fig. 4. Mapping the Microsoft Azure data (Cortez et al., 2017) to real benchmarks and plausible assumptions for choosing appropriate hosts [Stream workload performs best on E5-2650 and worst on E5-2630].

5. Performance evaluation

The placement policy assign a given workload trace (assuming inside a VM) to an appropriate host. The consolidation policy then ensures to transform the cluster state to an ideal one — which consume less energy due to fewer hosts in use. Consolidation is achieved through VM migrations (Ferreto et al., 2011). Various heuristic methods such as first fit (FF), best fit (BF), FillUp (Zakarya and Gillam, 2016), are then used to see the impact of allocation and consolidation policies over the energy consumption, prices, profits, and workload performance. Albeit, heuristics may not produce optimal results; however, they are demonstrated quite fast and quick as compared to optimal algorithms, particularly, for large-scale on-line problems (Ferreto et al., 2011). Besides these, we also discuss two different approaches to migrations: (i) intra-cluster — migrate VMs across different hosts that belong to a particular cluster, only; and inter-clusters — migrate VMs across different hosts that may belong to various clusters in different geographical areas.

5.1. Experimental results

Table 6 describes the results which we get in the simulated environment for various resource allocation and migration

Table 6
Energy consumption in kWh using the Google workload traces [the '+' sign denotes performance improvements or energy efficiency while the '−' sign represents loss in performance or energy costs].

Management policy	Energy consumption (kWh)					Savings (%)	Exe. time (h)	Performance gain/loss (%)	Number of migrations
	DC1	DC2	DC3	DC4	Total				
No migrations									
FF	478.45	601.56	872.43	998.11	2950.55	0	617.99	0	0
BF	423.56	577.89	865.33	910.23	2777.01	5.88	616.84	+0.19	0
FillUp	401.02	546.8	843.9	849.86	2641.58	10.47	617.68	+0.05	0
FillUp@LS	468.56	290.6	1021.67	750.67	2804.81	**14.2**	615.82	**+0.35**	0
Intra-cluster migrations									
FF	448.9	588.33	863.34	1004.09	2904.66	0	619.43	0	2934
BF	455.56	562.89	880.77	989.3	2888.52	0.56	619.02	+0.07	2459
FillUp	399.45	541.58	820.78	841.66	2603.47	10.37	618.03	+0.23	1902
FillUp@LS	349.59	286.8	1007.45	1045.7	2689.54	7.41	616.56	+0.46	3014
FollowMe@Location	502.06	284.55	1289.41	592.06	2668.08	8.14	621.73	−0.37	4367
FollowMe@Source	349.87	702.43	1288.77	503.67	2844.74	2.06	622.92	−0.56	3672
FollowMe@LS	481.34	282.88	1017.78	898.87	2680.87	7.7	622.01	−0.42	4703
Inter-clusters migrations									
FF	437.78	579.32	844.79	997.64	2859.53	0	616.42	0	3248
BF	451.33	560.01	867.08	968.65	2847.07	0.44	617.82	−0.23	3898
FillUp	391.32	540.68	821.56	878.12	2631.68	7.97	615.98	+0.07	2996
FillUp@LS	344.68	299.76	997.34	1021.98	2663.76	6.85	615.05	+0.22	3247
FollowMe@Location	500.76	299.43	1201.78	601.54	2603.51	8.95	625.78	−1.52	4993
FollowMe@Source	343.67	700.98	1189.67	513.78	2748.1	3.9	624.87	−1.37	4610
FollowMe@LS	445.76	281.99	1005.43	901.65	2634.83	7.86	623.34	−1.12	5193

techniques. When no migration is considered, then the placement policy "FillUp" could save approximately 10.47% more energy, along with marginal improvement in workload performance i.e. ∼0.05%, than the classical "FF" algorithm. However, when the allocation is aware of the energy prices and sources; then, energy savings increases (∼14.2%) along with performance further gains (∼0.35%). This demonstrates that workloads are largely placed in DC3 which has the lowest energy prices and PUE. Furthermore, DC2 is the most expensive one, therefore, placement is avoided there. When migrations are taken into account, variations in energy savings across different approaches to placement and migration policies is observed. Interestingly, if we migrate for better energy prices, then energy savings (8.14%) are greater than if we migrate for greener sources (2.06%) instead. This trade-off can be adjusted through the proposed "FollowMe@LS" policy with an approximate savings of 7.7% more than the classical "FF" allocation policy. These savings are possible at essential loss in workload performance (0.37%–1.52%); that might be non-trivial for certain kinds of application workloads. On average, the savings made by the proposed placement policy can be up to 15.26%, compared to the classical FF approach. Similar savings were also observed against the classical BF policy.

Furthermore, we observed that migrations can be expensive and it would be more economical not to migrate. This is in line and consistent with our previous findings in Zakarya and Gillam (2016). For example, for "BF" and "FillUp@LS" policies, migration could be approximately 4.02% and 6.24% expensive than no migration approach, respectively. Similarly, intra-clusters migrations are triggered more than intra-cluster migrations that could increase energy savings as high as 3.4%. However, for more tight packing (allocation policy i.e. "FillUp"), and considering migration costs; an approximate 1.08% loss in energy savings is expected in intra-clusters migrations, as well. Table 6 shows the total number of migrations for both migration opportunities. Figs. 5 and 6 show the total infrastructure energy cost and users monetary costs, respectively. Both figures demonstrate that "FollowMe@LS" balances the trade-off between moving workloads for sources and prices. It is possible to modify the proposed policies further to avoid expensive migrations in order to maintain user costs (Zakarya, 2018b). Moreover, additional constraints in placement and migration decisions, such as: (i) migrate only to a renewable with the least energy prices; or (ii) migrate only if target hosts/clusters are more economical (energy, performance and cost-efficient); and etc. This would certainly improve cost savings, in terms of energy bills, while maintaining the expected levels of workload performance.

5.2. Results discussion

In this section, we briefly describe the impact of various allocation and migration policies in the infrastructure energy consumption, given different prices and sources for energy. We also ascertain how workloads would affect the evaluated metrics. Table 7 shows various results which we obtained for another workload trace, offered from Microsoft Azure cloud (Cortez et al., 2017). These results are largely consistent with our previous outcomes; however, different impacts on energy consumption and performance can be seen very clearly. For example, when no migration are considered, then the proposed placement approach "FillUp@LS" can save ∼15.26% energy along with 0.74% improvements in performance. However, for migration scenarios, energy efficiency is negatively impacted (−3.44%), albeit with trivial performance improvements (1.98%). This is possibly caused due to the long-running behaviour of tasks in the workload type. Since, if workloads run for longer, they will absolutely consume more (even if they are placed on to resources powered by renewables). Moreover, if workloads run for longer, then, migration opportunities are decreased; thus resulting in lower energy efficiency. This is observed against the well-known classical heuristics such as FF, BF and FillUp.

Figs. 7 and 8 sketches a view of entire infrastructure energy bill and service costs paid by the customer for this particular workload type. These results demonstrate that migrations might be affective to decrease providers' energy bills; however, this has a negative impact on user costs i.e. service level agreements (SLAs). Violating SLAs may subsequently result in switching customers to other providers or, at least, penalties to the service provider. Both these options are not cost and revenue-effective for cloud providers, in particular, public providers. Besides these, and without migrations, various placement policies can result is various revenues for providers; and also customers. Again, the savings achieved through affective placement policies are significantly larger than the savings obtainable through migration techniques. Note that, in Table 7, "FollowMe@Location" policy

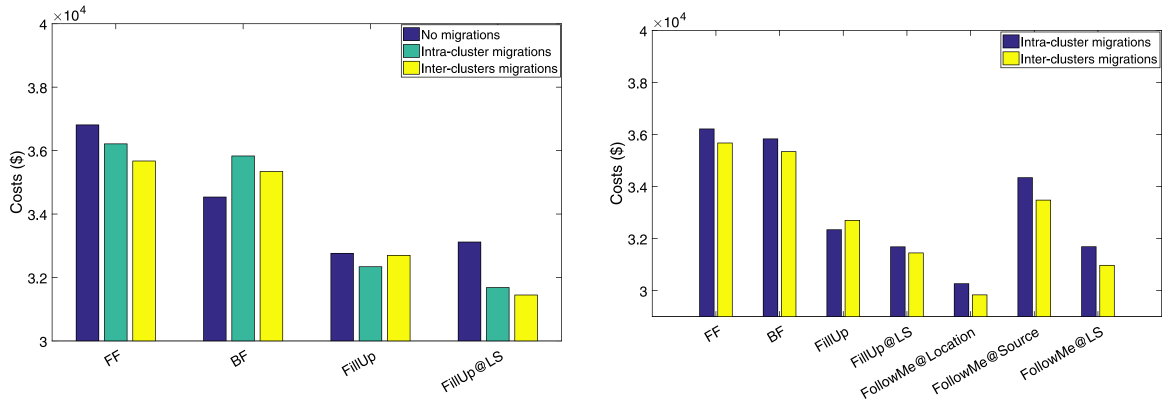

Fig. 5. Infrastructure costs — The corresponding PUE for each cluster was used to compute the energy used in non-computation infrastructure, such as cooling, and other facilities [left : no migration — right : with migrations].

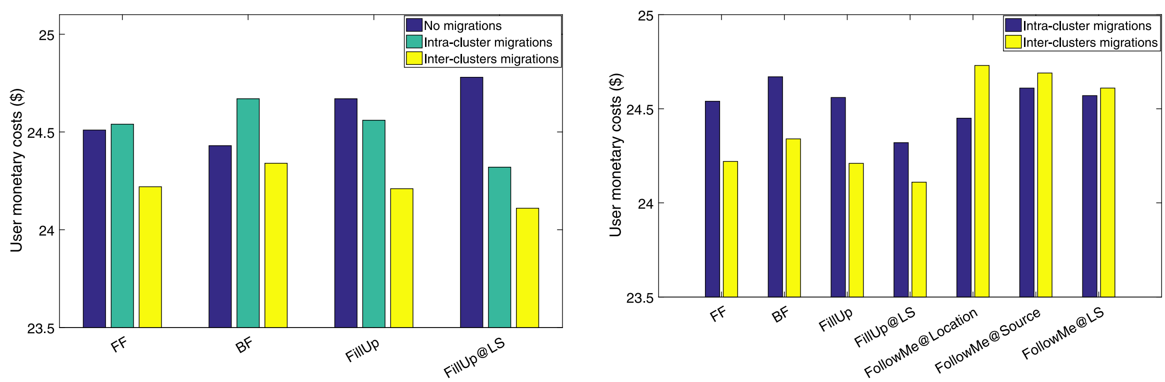

Fig. 6. User monetary costs [left : no migration — right : with migrations].

Table 7
Energy consumption in kWh using Microsoft Azure workload traces [the '+' sign denotes performance improvements or energy efficiency while the '−' sign represents loss in performance or energy costs].

Management policy	Energy consumption (kWh)					Savings (%)	Exe. time (h)	Performance gain/loss (%)	Number of migrations
	DC1	DC2	DC3	DC4	Total				
No migrations									
FF	621.78	783.67	993.56	1289.1	3688.11	0	1107.43	0	0
BF	602.79	742.89	1002.56	1183.68	3531.92	4.23	1103.45	+0.36	0
FillUp	599.88	709.65	1012.43	1134.2	3456.16	6.29	1111.9	−0.4	0
FillUp@LS	683.5	512.46	927.9	1001.34	3125.2	**15.26**	1099.21	+**0.74**	0
Intra-cluster migrations									
FF	598.43	790.34	901.4	999.87	3290.04	0	1121.89	0	2641
BF	575.22	701.45	973.59	1003.56	3253.82	1.1	1109.32	+1.12	2240
FillUp	644.98	700.22	990.49	1005.32	3341.01	−1.55	1112.67	+0.82	1684
FillUp@LS	600.45	655.89	1101.42	1045.54	3403.3	−3.44	1113.45	+0.75	2579
FollowMe@Location	534.68	510.53	1398.43	892.56	3336.2	−1.4	1108.11	+1.23	4290
FollowMe@Source	502.43	711.65	1267.89	703.21	3185.18	3.19	1107.3	+1.3	3543
FollowMe@LS	610.54	609.1	1288.76	738.87	3247.27	1.3	1110.9	+0.98	4401
Inter-clusters migrations									
FF	554.08	757.04	870.07	960.09	3141.28	0	1120.5	0	2571
BF	550.05	670.37	928.39	909.02	3057.84	2.66	1102.8	+1.98	3863
FillUp	630.71	661.54	956.65	969.3	3218.2	−2.45	1111.56	+0.8	2964
FillUp@LS	563.19	586.32	1062.51	990.29	3202.31	−1.94	1106.49	+1.25	2820
FollowMe@Location	518.39	447.75	1369.88	813.88	3149.9	−0.27	1122.67	−0.19	4654
FollowMe@Source	480.13	668.66	1247.34	647.88	3044.02	3.1	1113.9	+0.59	4410
FollowMe@LS	591.9	544.28	1249.98	702.8	3088.97	1.67	1103.9	+1.48	5105

migrates the workloads to geographical area with lower energy prices; that reduces energy efficiency; however, providers will still pay more for higher energy consumption (due to longer execution times i.e. performance degradation due to resource heterogeneity and longer migration durations). On the other hand, "FollowMe@Location" policy puts workloads on energy efficient clusters, which does not essentially indicates lower prices and high revenues for customers and providers both, due to existing trade-off between energy consumption and workload performance. Therefore, through mixing "FollowMe@Location" and "FollowMe@Source", benefits of both approaches could be achieved. For example, "FollowMe@LS" prefers to put or migrate workloads to hosts/clusters that has the least product of energy prices and renewables sources.

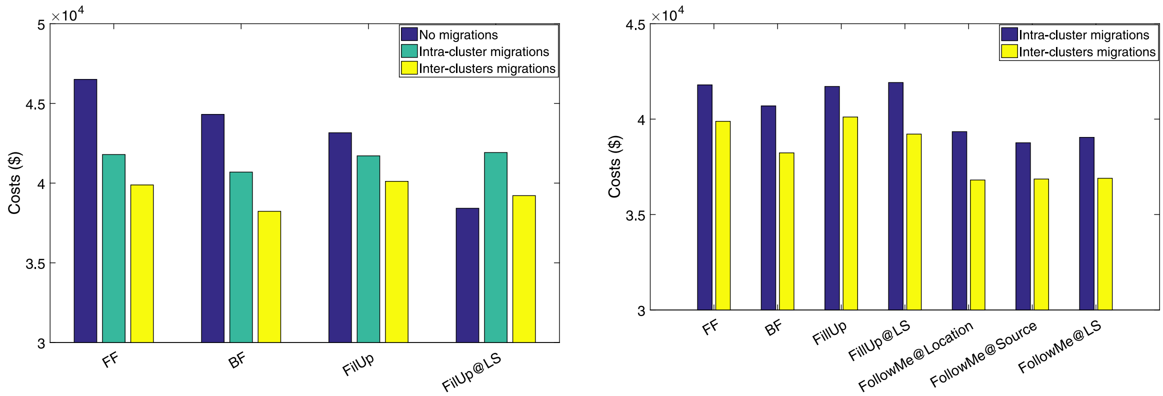

Fig. 7. Infrastructure costs — The corresponding PUE for each cluster was used to compute the energy used in non-computation infrastructure, such as cooling, and other facilities [left : no migration — right : with migrations].

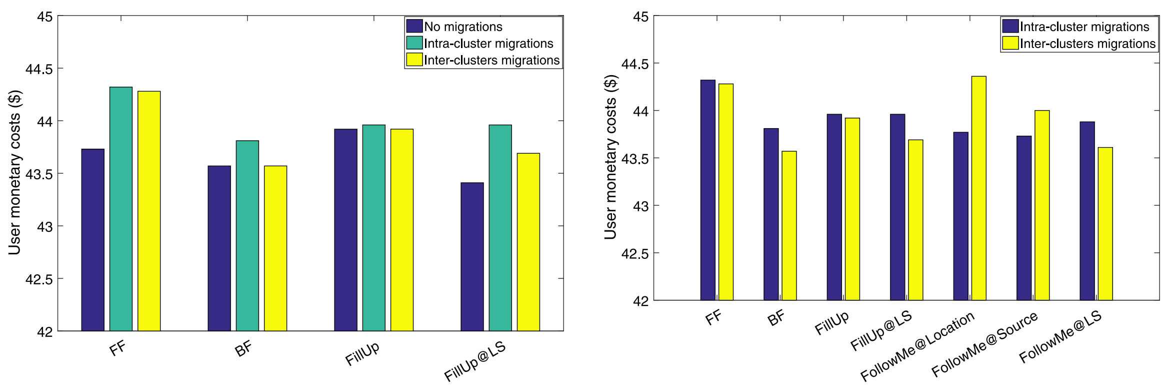

Fig. 8. User monetary costs [left : no migration — right : with migrations].

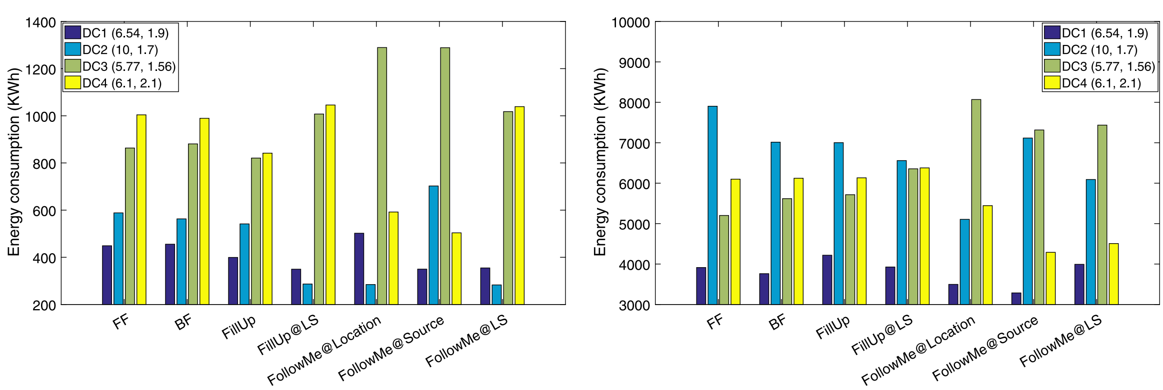

Fig. 9. Impact of various policies on energy consumption of different clusters — given different energy prices, sources and workloads [left : Google workload traces — right : Microsoft Azure traces].

Finally, Fig. 9 demonstrates the impact of various policies on energy consumption of different clusters given different energy prices, sources and workloads. For placement policies, various placement options have different energy consumption values. However, different workloads have non-trivial impacts on the infrastructure energy consumption. This creates a further gap for investigation, that what workload types should be run on which resources, or, geographical clusters. For example, certain applications e.g. real-time, might not be delayed for the availability of renewables in public clouds. Similarly, their migrations to other geographical areas may also not be feasible due to strict deadlines. Besides, it is a fact that long-running workloads over energy efficient clusters would consume more, subject to loss in performance. In the same way, short-running workload may frequently migrate (or trigger optimization of the datacenter state frequently), which may consume more, subject to number of migrations and their costs (Zakarya and Gillam, 2019a; Khan et al., 2020). However, energy efficiency and performance would differ if workloads utilizes their resource (such as CPU, memory, disk) differently. These workloads' challenges will essentially force service providers to think for other ways and placement/consolidation policies in order to manage their infrastructure located in different geographical areas and powered by various energy sources. These experiments and outcomes also denote the scalability of our approach in terms of large-scale heterogeneous systems, various workloads types and different parameters. However, the complexity of the algorithms will essentially increase with an increase in number of clusters (geographical locations), hosts within these clusters and users or demand for services. The time complexity (worst and average case) of the proposed algorithms are further described in Section 5.4.

5.3. Comparison with the closest rivals

Table 8 sketches a comparison of the proposed approach "FollowMe@LS" and state-of-the-art placement methods such as: workload shifting algorithm (WSA) (Xu and Buyya, 2020); energy and carbon efficient (ECE) VM placement (Khosravi, 2017); and cost and renewable aware with dynamic PUE (CRA-DP) placement (Khosravi et al., 2017). These methods account for energy prices and sources, but, migrations (both intra-cluster and inter-clusters) are not explored. Our approach could save significant amount of energy (approximately 8.7%–16.9% more than other approaches), but, with non-trivial performance loss (0.24%–0.69%). Albeit, we observed performance gains in certain scenarios; however, the mean value is the worst due to large variations among various iterations — as denoted by the largest standard deviation value i.e. 4.78. This degradation can be minimized further through incorporating some sort of migration control policies in order to avoid costly migrations. For example, migrations of relatively long-running VMs to more energy and/or performance efficient hosts might be preferred due to their higher chances of recovering their migration costs (Khan et al., 2019a). Larger energy savings will translate to higher profits, environmental friendly resources, service reputation, and revenue for service providers; however, performance loss will have impact on user satisfaction and providers revenues as well. Apart from these, we observed that if migrations are reduced to only intra-cluster methodology; then, our approach outperform all these methods in terms of various evaluation metrics i.e. energy efficiency, performance loss and user monetary costs. This is possibly due to lower migration overheads (short distances) and higher opportunities for consolidation (non-repeated migrations). Note that, repeated migrations might be expensive that should be controlled using appropriate mechanisms such as longer datacenter's reconfiguration periods (non-frequent runs of the optimization module).

5.4. Time and space complexity

In Alg. 1, initially all sources and locations are sorted with respect to prices taking $O(n^2)$ time in the worst case while $\Omega(n)$ in the best case. In this algorithm, the inner 'for loop' is for host "h" is within the outer 'for loop' for cluster 'c'; therefore, its worst case complexity is $O(n^2)$. Since, the VM list is in outer main loop which executes for each placement decision i.e. the inner loop executes. Therefore, the overall worst case execution time is given by: $T(n) = O(n) \times O(n^2) = O(n^3)$. Usually, sorting takes $O(n^2)$ time in the worst case, but the incorporated nested three loops will also take $O(n^3)$ time, in the worst case. Usually, the number of clusters or geographical locations are few enough and can be assumed as a constant; in which case, the average case complexity of Alg. 1 will be $O(n^2)$. The best case occurs when the VM is placed in the first attempt leading to $\Omega(n)$. After ignoring lower terms, we have the time complexity equal to $O(n^3)$. In Alg. 2, from steps 1 to 5 the worst case time complexity is $O(n)$. From steps 6 to 17, we have time complexity of $O(n^3)$. For steps 18 to 21, the time complexity is $O(n)$. As the higher time complexity is $O(n^3)$, so $T(n) = O(n^3)$. Again, assuming the number of clusters as constant, the average case complexity of Alg. 2 will be $O(n^2)$. The best case occurs when a VM is placed in its first attempt in the optimization phase. The best case complexity will be $\Omega(mn)$ as m is the best case for the optimization phase and n for the placement i.e. Alg. 1. The time complexity of the proposed algorithms will definitely increase with respect to workloads demand (number of users), capacity, availability, and usage of the IaaS resources. However, it is largely accepted, in the cloud scenarios, that heuristics are fast enough than optimal algorithms.

Space complexity denotes the total amount of memory that an algorithm requires for obtaining the desired outcomes for a specific input parameters (Yavari et al., 2019). The space complexity is strongly dependent, and exponentially increases/decreases, on the arrival rate of the VMs. This is due to the fact that each VM requires significant amount of memory (instance images) as well as enough memory to store the dirtying pages subject to different workload types. In the proposed placement and consolidation algorithms, the dominant variables are the number of VMs' and hosts' characteristics such as CPU and memory. No other details are much essential for all policies. Albeit, some additional memory is required for the power consumption model of each host that is specified by eleven values from 0 to 100% CPU utilization with an increment of ten percent (SPECpower benchmarks). Moreover, the space needed to optimize the state of the datacenter during each consolidation round will need enough space depending on the number of migratable VMs, and eligible hosts. Therefore, in the worst case, each algorithm requires ($n^2 + 11n$) additional units of memory. This exponentially increasing space complexity makes it infeasible and very difficult to validate the outcomes with huge numbers of simulated VMs, hosts and different types of workloads (Moges and Abebe, 2019; Homsi et al., 2019). In our previous work (Zakarya et al., 2020), we have described similar situations, in detail; and how we were able to conduct relatively large-scale simulated experiments on a small system through increasing memory slots and clearing heap space explicitly. Fig. 10 shows an exponential growth in memory usage in proportional to increasing the number of hosts and VMs. Note that, these values were obtained, in a single run, of the Google workload traces on an eight core CPU of 2.8 GHz and 16 GB of memory. The operating system overhead is also indicated by a vertical line over the *x-axis*. The duration of the experiment was 12 h and it has been observed that longer duration may potentially increase the space complexity of the proposed algorithms.

5.5. Summary of findings, results validity and limitations

In this paper, we proposed a placement policy "FillUp@LS" that puts appropriate workloads on appropriate clusters, according to energy sources and prices. Furthermore, three different consolidation policies "FollowMe@Location", "FollowMe@Source", and "FollowMe@LS" are proposed to migrate workloads, across geographically distributed clusters, in an energy, performance, cost effective way. These scheduling policies run in a distributed fashion — the global scheduler communicates with local schedulers to take appropriate workload execution decisions. In Section 5.5, we briefly explain our outcomes. talks over precision of the obtained results and describes limitations of our work.

Major findings: Through empirical evaluation using real workload traces from public service providers, we observed the following major findings:

- consolidation techniques are usually expensive and have negative impacts on the workload performance, and users' monetary costs;
- better VM allocation approaches could be more energy, performance, and cost-efficient than consolidating policies for certain kinds of workloads;
- migrating workloads can be ∼15.26% energy efficient; however, "FillUp@LS" (allocation) can be ∼28.58% energy efficient than the classical first fit policy;

Table 8
Comparison with the closest rivals accounting for both intra-cluster and inter-clusters migrations [the ± sign denotes the standard deviation across various runs −the lowest values are the best].

Approach	Evaluation metric		
	Energy consumption (kWh)	Performance (h)	Users' costs ($)
WSA (Xu and Buyya, 2020)	3689.54 ± 541.34	1099.8 ± 3.44	38.77
ECE (Khosravi, 2017)	3731.01 ± 202.7	1103.77 ± 0.75	42.54
CRA-DP (Khosravi et al., 2017)	3394.79 ± 186.93	1104.81 ± 2.91	41.40
FollowMe@Location	3267.65 ± 178.23	1109.21 ± 4.81	41.88
FollowMe@Source	3223.11 ± 121.89	1108.78 ± 4.21	41.55
FollowMe@LS	3099.33 ± 126.43	1107.45 ± 4.78	39.04

Table 9
Summary of the related work, closest to our work, with respect to various evaluation criteria [Exe. time refers to workload performance and Optimize denotes consolidation].

Approach	Methodology				Platform Multi-	Metric				
	Placement	Optimize	Renewables	Migration	clusters	Energy	Exe. time	Price	User costs	CO_2 footprints
Adnan et al. (2012)			✓		✓	✓		✓		
Chen et al. (2016)					✓	✓		✓		
Liu et al. (2012)			✓		✓	✓		✓		
Neglia et al. (2016)			✓	✓	✓	✓		✓		
Rossi et al. (2017)	✓	✓				✓				
Toosi et al. (2017)			✓		✓	✓		✓		
Beloglazov and Buyya (2015)	✓	✓				✓	✓		✓	
Cheng et al. (2014)			✓	✓	✓	✓				
Nguyen et al. (2017)	✓	✓				✓				
Khosravi et al. (2017)		✓	✓		✓	✓		✓		✓
Goiri et al. (2013)			✓	✓		✓				
Xu and Buyya (2020)			✓	✓	✓	✓				✓
Zakarya and Gillam (2019a)	✓	✓		✓		✓	✓		✓	
FillUp@LS	✓		✓		✓	✓	✓	✓	✓	
FollowMe@LS	✓	✓	✓	✓	✓	✓	✓	✓	✓	

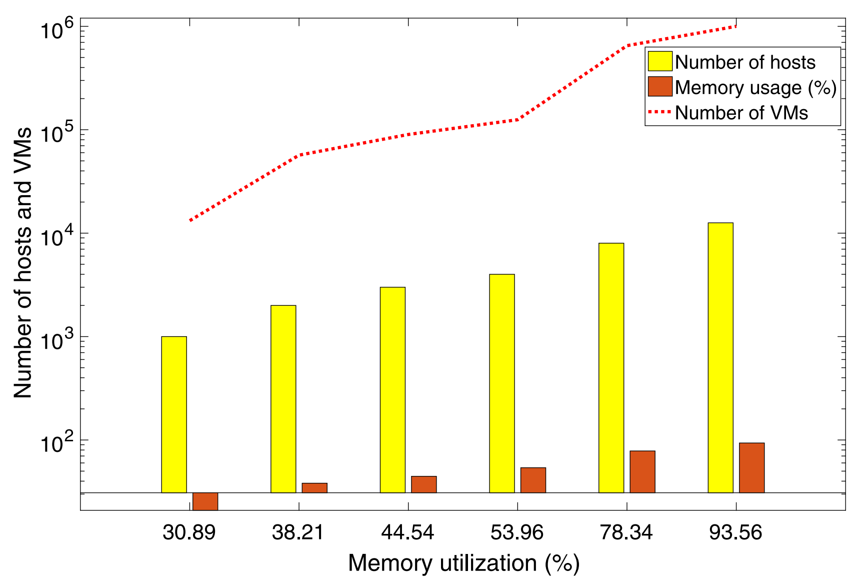

Fig. 10. Memory usage (%) with respect to increasing the total number of hosts and VMs.

- migrating only for the lowest costs i.e. "FollowMe@Location" or migrating to only renewable energy sources i.e. "FollowMe@Source" result in a trade-off among energy consumption, workload performance, and users' cost;
- migrating workloads using the proposed "FollowMe@LS" policy reduces approximately 23.89% energy consumption, and ∼19.66% users' costs while increasing ∼1.58% workload's performance, compared to the "no migration" approach; and
- resource management policies produces different outcomes which are strongly dependent on the workload type and how these workloads uses the IaaS resources.

Results validity: As demonstrated in our previous studies (Zakarya and Gillam, 2017a, 2019b), the developed version of the classical CloudSim simulator can produce approximately 98.63%–98.99% accurate and precise results as compared to a real IaaS private cloud. The accuracy is computed using appropriate statistical validation and verification approaches. The extended version of the CloudSim simulator which was used in our experimentation i.e. PerficientCloudSim was recently published in Zakarya et al. (2020); and is publicly available online at the GitHub repository. With this accuracy, it means that approximately ±1.01%–±1.37% error is expected in our simulated outcomes. With this accuracy, we can easily compute the expected errors in energy or performance efficiencies of various resource management policies.

For example, the resource management policy "FollowMe@LS", which is suggested approximately 23.89% more energy, 19.66% cost effective, and 1.58% more performance efficient than the "no migration" technique, could potentially save approximately 23.89 [±0.24% to ±0.33%] more energy, 19.66 [±0.12% to ±0.27%] users' costs, and is ∼1.58 [±0.016% to ±0.022%] more performance efficient than the "no migration" approach. Note that, the ±0.24–±0.33, ±0.12–±0.27, and ±0.016–±0.022 are approximately 1.01% and 1.37% of 23.89, 19.66, and 1.58, respectively.

Limitations: The above model has two shortcomings. (I) Cloud datacenters run heterogeneous applications with diverse resource usage, including not only the CPU, but also the memory, the disk, and the network. Those subsystems apart from the processor have been also reported making up a noticeable part of the total power consumption depending on the workload (Bircher and John, 2012). In order to avoid models that are specific for CPU-intensive applications, the impact on the power consumption of the rest of subsystems should be also considered. (II) Moreover, the VM CPU utilization is used to characterize the VM workload and to correlate the processor usage with the power consumption. However, the utilization is not the best indicator of the processor usage regarding its correlation with energy consumption, because applications with the same utilization can have different processor energy consumption depending on what instructions they are executing, as reported by Kansal et al. (2010).

We are aware of few issues with the proposed framework. First, when more and more VMs interact with the proposed scheduler and/or the consolidator then, due to delay in communication or network congestion, the system response might become slow. Slower response will essentially affect the system performance with respect to time and which may, subsequently, affect energy consumption, and users' costs. This issue is more likely to arise with increase in number of VMs. Secondly, the data maintained on each cluster node is a burden on it that keeps on maintaining and calculating statistical information regarding resource consumption, in addition, to performing its necessary task of job execution. Further, it also needs to update its information with the data server e.g. network area storage (NAS). Imagine hundreds or thousands of cluster nodes which are updating their information on NAS servers, periodically, which will itself generate a lot of traffic and, therefore, burden on the datacenter network. Further research is needed to account for these important issues. Besides these, in the United States the energy prices may vary with respect to usage and peak times. This research is limited to static energy prices in four different regions, as shown in Table 1. Further research and investigation is needed when these prices vary across different regions using an hourly or other unknown usage time periods (day and night). A study of robust deep learning based prediction techniques might be useful to estimate the migration and runtimes of workloads; and the heterogeneity of resources which can ensure workload independent energy, performance, and cost (EPC) aware resource/VM allocation and consolidation in IaaS clouds. Besides these, several other limitations of current work and further discussion around future research are presented in Section 7.

6. Related work

There is a huge amount of research going around to improve energy efficiency and performance for datacenters within the cloud research community. Energy efficiency for a datacenter can be achieved using a three level optimization i.e., software, hardware, and intermediate level, respectively (Zakarya, 2018a). The primary two methods used earlier for energy efficiency are VM consolidation (Ferreto et al., 2011) and Dynamic Voltage Frequency Scaling (DVFS) (Zakarya and Gillam, 2017a). More and more approaches have incorporated these two approaches in a dominant and significant way. Though, the drawback of these methods discuss that they are not good in situations when datacenters are overloaded. In overloaded datacenter scenarios, they do not function as required to improve energy and performance efficiencies due to the fact that an idle server still consumes 60% of the peak power consumption (Deng et al., 2014). Therefore, the saving made by CPU level approaches are far minimum than that of server level (Zakarya and Gillam, 2017b).

The resource management of multi-cloud infrastructure, geographically distributed, is discussed in many approaches in earlier works. A geographical-based load-balancing approach presented by Liu et al. (2012) uses renewable energy which helps to reduce the use of brown energy. An infrastructure presented by Toosi et al. (2017) tries to balance web based application loads across multiple datacenters where renewable energy is available and aims to reduce overall cost of the electricity. A method for energy and workload management is presented by Chen et al. (2016) where aim is to reduce energy and operational costs of the network. Another method presented by Adnan et al. (2012) focuses on dynamic workloads' deferral method targeted for multi-cloud enabling dynamic electricity prices at various locations as well as workload deadlines. A workload based scheduling method proposed by Neglia et al. (2016) discusses Markov chains to communicate workloads and renewable energy across geographically placed datacenters. These works presented targets to reduce the overall electricity costs but there is no consideration given to carbon footprints. Moreover, performance is not taken into account. The work presented by us, in this paper, focuses to take advantage of various scheduling and consolidation methods to increase energy efficiency, workload performance, decrease user costs etc.

In VM consolidation, the main aim is to consolidate VMs to fewer hosts in context to resource utilization and energy consumption — reduce energy consumption through increasing the utilization levels of fewer hosts. This allows more active hosts to run on low power and data is migrated from one host to another host. VM consolidation based OpenStack method proposed by Beloglazov and Buyya (2015) discusses energy efficiency. The approach saves power while the QoS is intact, where multiple heuristics are implemented based on VM consolidation. A combination of DVFS and VM consolidation based energy efficient cloud orchestrator is presented by Rossi et al. (2017). It allows to enhance the balance between power savage and application's performance. A real time simulation show significant savage of energy usage is observed by the presented orchestrator with slight amount of additional cost. Through application of VM consolidation for energy efficiency, a balance between migration time and energy usage specifically for datacenters placed in geographically distributed locations is achieved. A VM consolidation based work presented by Nguyen et al. (2017) discusses usage of multiple prediction based on local heuristics in order to enhance cloud datacenter's energy efficiency.

In current scenario, main focus and prediction is on resource utilization in order to figure out optimal place for VM consolidation to highlight under loaded or over loaded hosts within the datacenter. For distributed cloud infrastructure, a workload based migration and placement method is proposed by Cheng et al. (2014). It focuses on renewable energy availability while to improve performance of the datacenters within the distributed clouds. As compared with the work presented by us, it is only focused on batch workload and there is no attention provided towards carbon footprints. Cloud's resource usage and reduction in energy consumption for the datacenters can also be achieved through Graphics processing units (GPUs) (Silla et al., 2016).

An approach to analyse cluster equipped grouped together with virtual GPUs at remote stations by Iserte et al. (2016) show that use of GPUs helps to enhance resource usage and makes sure that energy constraints are met. The usage of GPUs in finance based application is presented by Varghese et al. (2015) which show that application's efficiency is obtained by using GPUs. Despite the work presented, our aim is to consider workloads distributed across the datacenters geographically placed at different time zones. The VM placement and consolidation mechanisms shown in our work are easily applicable over such heterogeneous infrastructures.

There is a significant research available where energy usage and carbon footprints are considered for datacenters within the cloud. A VM placement method by Khosravi et al. (2017) discusses energy reduction and carbon costs for datacenters placed geographically but with the limitation that all the locations are residing within the same country. A carbon footprint management approach by Doyle et al. (2013) discusses only load balancing but consideration over renewable energy is not focused. The Parasol and GreenSwitch scheme proposed by Goiri et al. (2013) takes a prototype system where dynamic scheduling is enabled for workloads and different energy sources are selected. Not like the work presented in Xu and Buyya (2020), this work also considers servers at the same location. In comparison to existing work presented, the approach in Xu and Buyya (2020), gives workload shifting in order to schedule workloads across various datacenters. The main objective of their work is to minimize overall carbon footprint as well as making sure that the average response time of the requests is intact. Along with these, their objective is also focused on geographically placed datacenters at different time zones having various carbon concentrations as well as renewable energy availability.

In Sheme et al. (2016), a new scheduling method for energy sources is formulated to enhance usage of renewable energy, and then considers reducing energy obtained from conventional grid and battery backup. The dynamic method encompasses to use grid power covering energy. The main advantage of this method is that it is evidently realistic to ponder supply of energy to a datacenter from the grid, though it has limitation to implement dynamic power. On contrary, it is also tried to optimize usage of battery by boosting low capacity of the batteries. The given algorithm gives high efficiency in case of renewable energy being efficiently and exhaustedly exploited by using workload scheduling. Furthermore, Liu et al. (2012) integrates workload management for datacenters by taking gains of efficiency made available by changing demand which exploits variations in time for electricity's price, renewable energy availability, and efficient cooling. There are two phases in the design i-e, first important feature is integrating three main silos of datacenters: IT, power and cooling. Secondly, a mix of theory, modelling, and implementation. The core of the design is focused on optimizing cost solved through workload management. The method depicts reduction of grid electricity consumption by approximately 60% having no impact over the quality of service provided by the applications.

All these works have relatively ignored the performance aspect of the datacenters' resources while moving or delaying workloads for later execution. Moreover, the impact of scheduling policies on datacenters' costs is relatively unexplored in the discussed literature. For example, in Toosi et al. (2017), the focus is on using renewable energy as a main source of energy for datacenters. It helps to reduce energy costs of brown energy but brings out challenges due to issues of highly discontinuous and unstable condition of wind and solar energy. Similarly, in Chen et al. (2016), the focus is on reduction of overall system energy cost of the system but it does not take into account the geographical location and time zones with respect to different electricity prices. Furthermore, in Adnan et al. (2012), a load balancing approach is implemented so that the energy cost is reduced using different time zones and locations for electricity prices using deferral method; but, it creates user dissatisfaction over dynamic price changes. In Neglia et al. (2016), a mean field method for load balancing among micro datacenters is used using renewable energy; but, it does not cover other sources of energy and their costs. Similarly, in Rossi et al. (2017), a DVFS-based VM consolidation approach is used for utilization of performance in order to reduce energy usage; however, in this approach resources are still underutilized. Table 9 describes summary of the related work. We believe, information in this table will help our readers to quickly identify gaps for further research, investigation and improvement.

7. Conclusions and future work

In this paper, we considered electricity sources and prices while provisioning the most economical resources to execute various applications and workloads in geographically distributed, heterogeneous, cloud datacenters. Furthermore, we assumed that various clusters are fuelled with different energy sources like coal, renewables; and the electricity prices offered at different locations vary with respect to time of the day and location to location. We proposed a placement approach "FillUp@LS" that puts workloads onto appropriate resources given the energy prices and energy, performance, cost efficiencies of the geographical clusters. Furthermore, we proposed three variants of the migration policy i.e. "FollowMe@Location", "FollowMe@Source", and "FollowMe@LS" that migrate workloads based on either energy prices, datacenter PUE, and both, respectively. Experimental results, using real workload traces, electricity prices, show ∼15.26% energy savings, ∼0.53%–∼19.66% reductions in service monetary costs, and ∼1.58% improvements in applications' performance, against the FF heuristic algorithm. The estimated error in our results, due to simplification and simulation models, is suggested to be ±1.01% to ±1.37% (Zakarya et al., 2020). Furthermore, we observed that various applications and workloads may perform quite differently, due to heterogeneity of IaaS resources, energy sources, and prices; therefore, leading to variations in costs, revenues and energy consumptions.

We believe, the topic investigated in this research further suggests investigation and deep analysis of public cloud workloads that are significantly different from private clusters and distributed platforms (Feitelson, 2015). This is essential to justify the ideas of: (i) delaying workloads for the availability of the renewables; and (ii) shifting and migrating them to appropriate locations so that energy efficiency along with performance gains and higher profit can be obtained. In respect of (i), workloads having strict deadlines, and/or need quick response times (real-time applications or cloud services) cannot be delayed. However, batch processing of certain applications could be delayed and scheduled to run at later times; which might be beneficial with respect to energy consumption, providers revenue, user's costs, and ecological impacts. In respect of (ii), migration of workloads should account for certain aspects such as user's location, mobility, impact on the workloads' performance and, most importantly, the energy savings and providers' revenue achievable through migrating them while accounting for their migration costs (Zakarya et al., 2020). Besides workloads classification and prediction, different approaches to design workloads placement policies, schedulers, and consolidators, such as single (centralized) and multiple (distributed and/or hierarchical — decentralized), also need further investigation for energy, performance, and cost efficiencies in large-scale heterogeneous cluster environments.

Characterizing workloads will also need significant efforts over the prediction mechanisms. However, a prediction system might suffer from at least one, and occasionally all, of the several issues including: (i) the need of considerable amount of memory and complex data structures to store the history of users' jobs; (ii) the need of a complex prediction approach; and (iii) significant computational and storage (storage area network) overheads for maintaining the jobs history; and searching it for exact match and reaching an appropriate placement decision. However, Tsafrir et al. (2007) demonstrated that a very simple predictor can do an excellent job. For example, their outcomes obtained through designing a very simple prediction algorithm — the average runtime of the two most recently submitted (and terminated) jobs by the same user; which is easy to implement and almost costs no computational or storage overheads. The findings suggest that the predictor's successful capability is due to the fact that it only focuses on recent jobs (requiring less memory and storage capacity), in contrast to the previously proposed prediction methods that have largely focused on similarity in terms of job numerous characteristics such as runtimes, resource requirements, submitting users, workload types, and resource usage (Cortez et al., 2017; Calheiros et al., 2015; Tumanov et al., 2016; Amvrosiadis et al., 2017). In the future, machine learning based prediction techniques can be integrated with our proposal to trigger appropriate energy, performance, and cost effective workload placement, resource allocation and migration decisions. Finally, more accurate and reasonable models for energy consumption, migrations costs, performance loss (in particular co-located VMs that compete for similar resources) should be considered for further research and investigation.

CRediT authorship contribution statement

Hashim Ali: Conceptualization, Methodology, Investigation. **Muhammad Zakarya:** Supervision, Conceptualization, Writing - original draft, Software. **Izaz Ur Rahman:** Visualization, Data curation. **Ayaz Ali Khan:** Visualization, Writing, Formal analysis. **Rajkumar Buyya:** Validation, Project administration, Writing - review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work is supported, in part, by the Abdul Wali Khan University, Mardan (AWKUM), Islamic Republic of Pakistan and, in part, by the Higher Education Commission (HEC), Pakistan. Moreover, this research is also supported, in part, by an Australian Research Council (ARC) Discovery Project.

References

Adnan, Muhammad Abdullah, Sugihara, Ryo, Gupta, Rajesh K., 2012. Energy efficient geographical load balancing via dynamic deferral of workload. In: 2012 IEEE Fifth International Conference on Cloud Computing. IEEE, pp. 188–195.

Amvrosiadis, George, Park, Jun Woo, Ganger, Gregory R, Gibson, Garth A, Baseman, Elisabeth, DeBardeleben, Nathan, 2017. Bigger, Longer, Fewer: What Do Cluster Jobs Look Like Outside Google? Technical Report CMU-PDL-17-104, Carnegie Mellon Univedrsity.

Beloglazov, Anton, Buyya, Rajkumar, 2012. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. Concurr. Comput.: Pract. Exper. 24 (13), 1397–1420.

Beloglazov, Anton, Buyya, Rajkumar, 2015. Openstack neat: a framework for dynamic and energy-efficient consolidation of virtual machines in openstack clouds. Concurr. Comput.: Pract. Exper. 27 (5), 1310–1333.

Bircher, William Lloyd, John, Lizy K., 2012. Complete system power estimation using processor performance events. IEEE Trans. Comput. 61 (4), 563–577.

Calheiros, Rodrigo N, Masoumi, Enayat, Ranjan, Rajiv, Buyya, Rajkumar, 2015. Workload prediction using arima model and its impact on cloud applications' qos. IEEE Trans. Cloud Comput. 3 (4), 449–458.

Calheiros, Rodrigo N, Ranjan, Rajiv, Beloglazov, Anton, De Rose, César AF, Buyya, Rajkumar, 2011. Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Softw. - Pract. Exp. 41 (1), 23–50.

Chen, Tianyi, Zhang, Yu, Wang, Xin, Giannakis, Georgios B., 2016. Robust geographical load balancing for sustainable data centers. In: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, pp. 3526–3530.

Cheng, Dazhao, Jiang, Changjun, Zhou, Xiaobo, 2014. Heterogeneity-aware workload placement and migration in distributed sustainable datacenters. In: 2014 IEEE 28th International Parallel and Distributed Processing Symposium. IEEE, pp. 307–316.

Cortez, Eli, Bonde, Anand, Muzio, Alexandre, Russinovich, Mark, Fontoura, Marcus, Bianchini, Ricardo, 2017. Resource central: Understanding and predicting workloads for improved resource management in large cloud platforms. In: Proceedings of the 26th Symposium on Operating Systems Principles. ACM, pp. 153–167.

Deng, Xiang, Wu, Di, Shen, Junfeng, He, Jian, 2014. Eco-aware online power management and load scheduling for green cloud datacenters. IEEE Syst. J. 10 (1), 78–87.

Doyle, Joseph, Shorten, Robert, O'Mahony, Donal, 2013. Stratus: Load balancing the cloud for carbon emissions control. IEEE Trans. Cloud Comput. 1 (1), http://dx.doi.org/10.1109/TCC.2013.4, 1–1.

Feitelson, Dror G., 2015. Workload Modeling for Computer Systems Performance Evaluation. Cambridge University Press.

Ferreto, Tiago C., Netto, Marco A S, Calheiros, Rodrigo N., De Rose, César A F, 2011. Server consolidation with migration control for virtualized data centers. Future Gener. Comput. Syst. 27 (8), 1027–1034.

Goiri, Íñigo, Katsak, William, Le, Kien, Nguyen, Thu D, Bianchini, Ricardo, 2013. Parasol and greenswitch: Managing datacenters powered by renewable energy. ACM SIGPLAN Not. 48 (4), 51–64.

Homsi, Soamar, Quan, Gang, Wen, Wujie, Chapparo-Baquero, Gustavo A, Njilla, Laurent, 2019. Game theoretic-based approaches for cybersecurity-aware virtual machine placement in public cloud clusters. In: 2019 19th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID). IEEE, pp. 272–281.

Iserte, Sergio, Prades, Javier, Reaño, Carlos, Silla, Federico, 2016. Increasing the performance of data centers by combining remote gpu virtualization with slurm. In: 2016 16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid). IEEE, pp. 98–101.

Kansal, Aman, Zhao, Feng, Liu, Jie, Kothari, Nupur, Bhattacharya, Arka A, 2010. Virtual machine power metering and provisioning. In: Proceedings of the 1st ACM Symposium on Cloud Computing. ACM, pp. 39–50.

Khan, Ayaz Ali, Zakarya, Muhammad, Buyya, Rajkumar, Khan, Rahim, Khan, Mukhtaj, Rana, Omer, 2019a. An energy and performance aware consolidation technique for containerized datacenters. IEEE Trans. Cloud Comput..

Khan, Ayaz Ali, Zakarya, Muhammad, Khan, Rahim, 2019b. Energy-aware dynamic resource management in elastic cloud datacenters. Simul. Model. Pract. Theory 92, 82–99.

Khan, Ayaz Ali, Zakarya, Muhammad, Khan, Rahim, 2019c. H^2 – a hybrid heterogeneity aware resource orchestrator for cloud platforms. IEEE Syst. J. 13 (4), 3873–3876.

Khan, Ayaz Ali, Zakarya, Muhammad, Khan, Rahim, Rahman, Izaz Ur, Khan, Mukhtaj, et al., 2020. An energy, performance efficient resource consolidation scheme for heterogeneous cloud datacenters. J. Netw. Comput. Appl. 150, 102497.

Khosravi, Atefeh, 2017. Energy and Carbon-Efficient Resource Management in Geographically Distributed Cloud Data Centers (PhD thesis).

Khosravi, Atefeh, Andrew, Lachlan L.H., Buyya, Rajkumar, 2017. Dynamic vm placement method for minimizing energy and carbon cost in geographically distributed cloud data centers. IEEE Trans. Sustain. Comput. 2 (2), 183–196.

Koronen, Carolina, Åhman, Max, Nilsson, Lars J., 2020. Data centres in future European energy systems—energy efficiency, integration and policy. Energy Effic. 13 (1), 129–144.

Lebre, Adrien, Pastor, Jonathan, Simonet, Anthony, Südholt, Mario, 2019. Putting the next 500 VM placement algorithms to the acid test: The infrastructure provider viewpoint. IEEE Trans. Parallel Distrib. Syst. 30 (1), 204–217.

Liu, Zhenhua, Chen, Yuan, Bash, Cullen, Wierman, Adam, Gmach, Daniel, Wang, Zhikui, Marwah, Manish, Hyser, Chris, 2012. Renewable and cooling aware workload management for sustainable data centers, In: Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems, pp. 175–186.

Liu, Haikun, Jin, Hai, Xu, Cheng-Zhong, Liao, Xiaofei, 2011. Performance and energy modeling for live migration of virtual machines. Cluster Comput. 249–264.
Liu, Zhenhua, Wierman, Adam, Chen, Yuan, Razon, Benjamin, Chen, Niangjun, 2013. Data center demand response: Avoiding the coincident peak via workload shifting and local generation. Perform. Eval. 70 (10), 770–791.
Moges, Fikru Feleke, Abebe, Surafel Lemma, 2019. Energy-aware VM placement algorithms for the openstack neat consolidation framework. J. Cloud Comput. 8 (1), 2.
Neglia, Giovanni, Sereno, Matteo, Bianchi, Giuseppe, 2016. Geographical load balancing across green datacenters: A mean field analysis. ACM SIGMETRICS Perform. Eval. Rev. 44 (2), 64–69.
Nguyen, Trung Hieu, Di Francesco, Mario, Yla-Jaaski, Antti, 2017. Virtual machine consolidation with multiple usage prediction for energy-efficient cloud data centers. IEEE Trans. Serv. Comput..
O'Loughlin, John, 2018. A Workload-Specific Performance Brokerage for Infrastructure Clouds (PhD thesis). University of Surrey.
O'Loughlin, John, Gillam, Lee, 2014. Performance evaluation for cost-efficient public infrastructure cloud use. In: International Conference on Grid Economics and Business Models. Springer, pp. 133–145.
O'Loughlin, John, Gillam, Lee, 2016. Sibling virtual machine co-location confirmation and avoidance tactics for public infrastructure clouds. J. Supercomput. 72 (3), 961–984.
Reiss, Charles, Tumanov, Alexey, Ganger, Gregory R, Katz, Randy H, Kozuch, Michael A, 2012. Heterogeneity and dynamicity of clouds at scale: Google trace analysis. In: Proceedings of the Third ACM Symposium on Cloud Computing. ACM, p. 7.
Reiss, Charles, Wilkes, John, Hellerstein, Joseph L., 2011. Google Cluster-Usage Traces: Format+ Schema. Technical Report, Google Inc., Mountain View, CA, USA.
Rossi, Fábio D, Xavier, Miguel G, De Rose, César AF, Calheiros, Rodrigo N, Buyya, Rajkumar, 2017. E-eco: Performance-aware energy-efficient cloud data center orchestration. J. Netw. Comput. Appl. 78, 83–96.
Sharma, Yogesh, Si, Weisheng, Sun, Daniel, Javadi, Bahman, 2019. Failure-aware energy-efficient VM consolidation in cloud computing systems. Future Gener. Comput. Syst. 94, 620–633.
Shehabi, A, Smith, SJ, Horner, N, Azevedo, I, Brown, R, Koomey, J, Masanet, E, Sartor, D, Herrlin, M, Lintner, W, 2016. United States Data Center Energy Usage Report. Lawrence Berkeley National Laboratory, Berkeley, California. LBNL-1005775, p. 4.
Sheme, Enida, Stolf, Patricia, Da Costa, Georges, Pierson, Jean-Marc, Frashëri, Neki, 2016. Efficient energy sources scheduling in green powered datacenters: A cloudsim implementation.
Silla, Federico, Prades, Javier, Iserte, Sergio, Reano, Carlos, 2016. Remote gpu virtualization: Is it useful? In: 2016 2nd IEEE International Workshop on High-Performance Interconnection Networks in the Exascale and Big-Data Era (HiPINEB). IEEE, pp. 41–48.
Tchana, Alain, De Palma, Noel, Safieddine, Ibrahim, Hagimont, Daniel, 2016. Software consolidation as an efficient energy and cost saving solution. Future Gener. Comput. Syst. 58, 1–12.
Toosi, Adel Nadjaran, Qu, Chenhao, de Assunção, Marcos Dias, Buyya, Rajkumar, 2017. Renewable-aware geographical load balancing of web applications for sustainable data centers. J. Netw. Comput. Appl. 83, 155–168.
Tsafrir, Dan, Etsion, Yoav, Feitelson, Dror G., 2007. Backfilling using system-generated predictions rather than user runtime estimates. IEEE Trans. Parallel Distrib. Syst. 18 (6), 789–803.
Tumanov, Alexey, Jiang, Angela, Park, Jun Woo, Kozuch, Michael A, Ganger, Gregory R, 2016. JamaisVu: Robust Scheduling with Auto-Estimated Job Runtimes. Technical Report CMU-PDL-16-104, Carnegie Mellon University.
US Energy Information Administration, 2019. What is US electricity generation by energy source?
Varghese, Blesson, Prades, Javier, Reaño, Carlos, Silla, Federico, 2015. Acceleration-as-a-service: Exploiting virtualised GPUs for a financial application. In: 2015 IEEE 11th International Conference on E-Science. IEEE, pp. 47–56.
Verma, Abhishek, Pedrosa, Luis, Korupolu, Madhukar, Oppenheimer, David, Tune, Eric, Wilkes, John, 2015. Large-scale cluster management at google with borg. In: Proceedings of the Tenth European Conference on Computer Systems. ACM, p. 18.
Xu, Minxian, Buyya, Rajkumar, 2020. Managing renewable energy and carbon footprint in multi-cloud computing environments. J. Parallel Distrib. Comput. 135, 191–202.
Xu, Fei, Liu, Fangming, Jin, Hai, 2016. Heterogeneity and interference-aware virtual machine provisioning for predictable performance in the cloud. IEEE Trans. Comput. 65 (8), 2470–2483.
Xu, Fei, Liu, Fangming, Jin, Hai, Vasilakos, Athanasios V., 2014. Managing performance overhead of virtual machines in cloud computing: A survey, state of the art, and future directions. Proc. IEEE 102 (1), 11–31.
Yavari, Maede, Rahbar, Akbar Ghaffarpour, Fathi, Mohammad Hadi, 2019. Temperature and energy-aware consolidation algorithms in cloud computing. J. Cloud Comput. 8 (1), 1–16.
Zakarya, Muhammad, 2018a. Energy, performance and cost efficient datacenters: A survey. Renew. Sustain. Energy Rev. 94, 363–385.
Zakarya, Muhammad, 2018b. An extended energy-aware cost recovery approach for virtual machine migration. IEEE Syst. J. 13 (2), 1466–1477.
Zakarya, Muhammad, Gillam, Lee, 2016. An energy aware cost recovery approach for virtual machine migration. In: International Conference on the Economics of Grids, Clouds, Systems, and Services. Springer, pp. 175–190.
Zakarya, Muhammad, Gillam, Lee, 2017a. Energy and Performance Aware Resource Management in Heterogeneous Cloud Datacenters (PhD thesis). University of Surrey.
Zakarya, Muhammad, Gillam, Lee, 2017b. Energy efficient computing, clusters, grids and clouds: A taxonomy and survey. Sustain. Comput. Inform. Syst. 14, 13–33.
Zakarya, Muhammad, Gillam, Lee, 2019a. Managing energy, performance and cost in large scale heterogeneous datacenters using migrations. Future Gener. Comput. Syst. 93, 529–547.
Zakarya, Muhammad, Gillam, Lee, 2019b. Modelling resource heterogeneities in cloud simulations and quantifying their accuracy. Simul. Model. Pract. Theory 94, 43–65.
Zakarya, Muhammad, Gillam, Lee, Khan, Ayaz Ali, Rahman, Izaz Ur, 2020. Perficientcloudsim: a tool to simulate large-scale computation in heterogeneous clouds. J. Supercomput. 1–55.

Hashim Ali is an Assistant Professor at the Department of Computer Science, Abdul Wali Khan University Mardan, Pakistan. He is currently a Ph.D. student in the same department. He completed his M.Phil (MS) in computer science from COMSATS Institute of Information Technology (CIIT), Abbotabad, Pakistan. His research interests include cloud computing, software testing, Agile processes, energy and performance efficient distributed systems and enterprise systems. He is proficient in computer systems both theoretically and practically.

Muhammad Zakarya received the Ph.D. degree in computer science from the University of Surrey, Guildford, U.K. He is currently a Lecturer with the Department of Computer Science, Abdul Wali Khan University Mardan, Pakistan. His research interests include cloud computing, mobile edge clouds, performance, energy efficiency, algorithms, and resource management. He has deep understanding of the theoretical computer science and data analysis. Furthermore, he also owns deep understanding of various statistical techniques which are, largely, used in applied research. He is an Associate Editor for the IEEE Access Journal and has served as TPC member in various international level conferences and workshops.

Izaz Ur Rahman received the Ph.D. degree in computer science from the Department of Electronic and Computer Engineering, Brunel University, London, UK. He is currently an Assistant Professor with the Department of Computer Science, Abdul Wali Khan University Mardan, Pakistan. His research interest includes power systems, optimization algorithms such as particle swarms, fuzzy logic, distributed systems, machine learning and artificial intelligence. He is teaching AI and fuzzy systems to graduate and undergraduate level students.

Ayaz Ali Khan holds a Ph.D. degree from the Department of Computer Science, Abdul Wali Khan University Mardan, Pakistan. He completed his M.Phil (MS) in computer science from COMSATS Institute of Information Technology (CIIT), Islamabad, Pakistan. His area of research includes energy-aware and performance-efficient scheduling, resource allocation, placement and management, at datacenter level. Moreover, he has enough knowledge of distributed systems, optimization, game theory and computer programming.

Rajkumar Buyya is a Fellow of IEEE and Life Member of ACM, Professor of Computer Science and Software Engineering, Future Fellow of the Australian Research Council, and Director of the Cloud Computing and Distributed Systems (CLOUDS) Laboratory, School of Computing and Information Systems, at the University of Melbourne, Australia. His research interests include cloud, grid, distributed, and parallel computing. R. Buyya has a Ph.D. in computer science from Monash University. He is also serving as the founding CEO of Manjrasoft Pty Ltd., a spin-off company of the University, commercializing its innovations in Cloud Computing. He has co-founded five IEEE/ACM international conferences: CCGrid, Cluster, Grid, e-Science, and UCC (Utility and Cloud Computing) and served as the Chair of their inaugural meetings. He has presented over 400 invited talks (keynotes, tutorials, and seminars) on his vision on IT Futures and advanced computing technologies at international conferences and institutions in Asia, Australia, Europe, North America, and South America. For further information on Dr. Buyya, please visit: http://www.buyya.com.