Energy-aware QoS-based dynamic virtual machine consolidation approach based on RL and ANN

Noname manuscript No.
(will be inserted by the editor)

Energy-Aware QoS-based Dynamic Virtual Machine Consolidation Approach based on RL and ANN

Mahshid Rezakhani ·
Nazanin Sarrafzadeh-Ghadimi ·
Reza Entezari-Maleki ✉ ·
Leonel Sousa ·
Ali Movaghar

Received: date / Accepted: date

Abstract One of the most challenging problems in cloud datacenters is the degradation of performance and energy efficiency due to the overutilization of hosts and their exposition to excessive workload. Virtual machine (VM) consolidation and migration from one host to another are strategies that have been proven to successfully bring about performance improvements and energy efficiency. These schemes help in energy optimization by moving VMs experiencing difficulty functioning on an overloaded host to another host. Similarly, by migrating VMs from an underloaded host and consolidating them, unnecessary resources have a chance to be shut down. This makes clear why the accurate detection of overloaded and underloaded hosts is of fundamental importance

Mahshid Rezakhani
Department of Computer Engineering, Sharif University of Technology, Tehran, Iran. E-mail: mrezakhani@ce.sharif.edu

Nazanin Sarrafzadeh-Ghadimi
School of Computer Engineering, Iran University of Science and Technology, Tehran, Iran. E-mail: nazanin_sarrafzade@alumni.iust.ac.ir

Reza Entezari-Maleki (corresponding author)
School of Computer Engineering, Iran University of Science and Technology, Tehran, Iran, and School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran, Iran, and INESC-ID, Instituto Superior Técnico, Universidade de Lisboa, Lisbon, Portugal. E-mail: entezari@iust.ac.ir

Leonel Sousa
INESC-ID, Instituto Superior Técnico, Universidade de Lisboa, Lisbon, Portugal. E-mail: las@inesc-id.pt

Ali Movaghar
Department of Computer Engineering, Sharif University of Technology, Tehran, Iran. E-mail: movaghar@sharif.edu

when energy consumption, quality of services, and service level agreements are targeted. In this paper, an energy-aware QoS-based consolidation algorithm is proposed to dynamically manage VMs in cloud datacenters. The proposed algorithm applies reinforcement learning and artificial neural networks. The first method is used to select a suitable VM for migration, while the latter helps to predict the future state of hosts and detect overloaded and underloaded hosts. We simulated the proposed algorithm using the CloudSim framework and compared it to the baselines and state-of-the-art algorithms. The results show that the proposed approach surpasses other methods in what concerns both performance and energy efficiency.

Keywords cloud datacenters · virtual machine · live migration · energy efficiency · reinforcement learning · artificial neural network.

1 Introduction

Owing to the ever-growing applicability and prevalence of data processing and storage, along with the ubiquity of internet-based services in recent years, access to resources and services has become facilitated. Thus, cloud computing is used so that organizations' needs, such as the storage of resources, computing, and accessing a variety of services can be outsourced. This means that instead of investing unreasonable amounts of money in a private computing center and paying for its maintenance, the cloud can be used as a more economical solution [1].

As an increasing number of users are getting involved in cloud technology, efficient use and proper management of cloud servers to manage resources and reduce energy consumption have become crucial [2]. Data centers require a lot of energy to meet the ever-growing needs of users. Computing is one of the subsystems that if optimized, can be of great help to energy efficiency. One of the most promising approaches to reaching this goal is dynamic VM consolidation, which gathers virtual machines in such a way that the number of required servers is minimized [3]. Since servers are incapable of fully exploiting their resources, if VMs are migrated and consolidated on a single host, the remaining computing systems can be shut down leading to a considerable amount of energy being saved [2, 4]. Studies show that organizations and research institutes have carried out extensive research in this area over the past few years [5]. This scheme, however, introduces some challenges regarding the performance of hosts and the quality of service (QoS) [6–9]. For instance, if the abundance of VMs incapacitates the server from providing sufficient resources and handling the abrupt fluctuations in VMs, a large delay will be imposed on the system, which diminishes the QoS and, consequently, raises the probability of service level agreement (SLA) violations [10–12]. Therefore, there is a need for a strategy to obtain an energy-SLA balance and offer high-quality services.

In general, the proposed approach, in this paper, tries to answer the following questions to solve this problem: (1) which hosts are overloaded? (2) which hosts are underloaded? (3) which VM should be selected for migration? The

appropriate answer to these questions leads to efficient resource management, reduction of energy consumption, and higher QoS. Various VM consolidation approaches for efficient energy consumption have been put forward. However, performance degradation stemming from the huge number of migrations still remains an issue in those methods [13]. The proposed approach pays extra attention to the number of migrations in addition to energy efficiency and QoS. To achieve this objective, artificial neural networks (ANNs) are initially used to intelligently predict the future state of hosts and to find overloaded or underloaded servers. ANNs help us in gaining insights into future resource and energy requirements, thus assisting us in the efficient management of these resources. RL solutions work based on the observation of the environment and take actions according to the knowledge they have gained through trial and error. It helps them identify complex relations between the parameters of the problem, thus achieving more reliable results. This is why the selection of the proper VM for migration is performed using reinforcement learning (RL). We evaluated the proposed approach using PlanetLab's real workload and compared it to other approaches. The results show that the proposed approach reduces energy consumption and the number of SLA violations by 22% and 94%, respectively, thus significantly improving the QoS.

The rest of this paper is organized as follows. Section 3 reviews related research. Section 2 provides background information on ANN and RL. In Section 4, the proposed approach is introduced, and it is evaluated in Section 5, through numerical results. Finally, Section 6 concludes the paper and provides some guidelines for future work.

2 Background information

This section provides background information on ANNs and RL. For more detailed information please see [14–17].

2.1 Artificial Neural Networks

Artificial neural networks (ANNs), simply called neural networks, are massive computing systems, inspired by the structure of a biological brain, consisting of a huge number of simple units known as nodes and their means of connectivity called edges. The way they are modeled enables them to learn from the examples provided by independent datasets without the need for any specific programming [14], which helps them predict and execute functions similar to the way a human brain does. This outstanding property of neural networks called *self-learning* makes it possible to learn through experience and apparently unrelated data. Therefore, they are adaptable to various contexts such as predictive modeling, data validation, anomaly detection in data, speech recognition, pattern processing, network traffic checking, and other problems even too convoluted for humans or computers to solve. Fig. 1 shows the structure

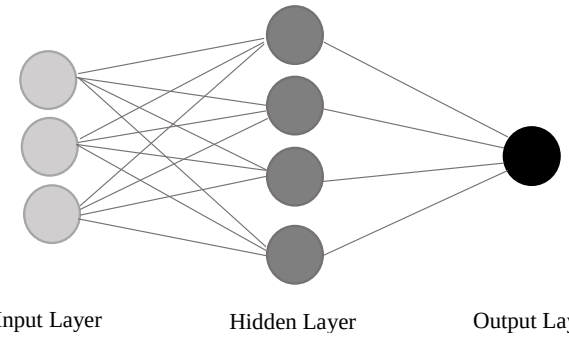


Fig. 1 The schematic view of a three-layered neural network

of a neural network. As it is shown in Fig. 1, ANNs generally encompass three layers starting from the input layer, a hidden layer, and finally, the output layer. These networks are trained using a workload consisting of input and output pairs. In the first step, the workload is given to the nodes as inputs, which is then modified by non-linear functions and the weights of edges, creating the outputs of each layer. After that, these outputs are compared with optimal results (target outputs), and the error is determined by the difference between these two values, which is also used to adjust the weights of the edges in the final step. This procedure is repeated with all datasets until the entire network is trained, the weights reach the desired values and the outputs achieve an acceptable error. Once the training has been completed, multiple datasets are given to the network for evaluation. Then the training stops and from this moment on, actual inputs are given to the input layer to predict the outputs. In other words, the network is now able to predict and model the results and generate a suitable output for each new input data [18]. This indicates that by the appropriate adjustment of variables and selection of datasets more accurate results are produced as the network is well-trained.

As mentioned earlier, learning takes place based on the comparison of the predicted outcomes and target values. For this purpose, *feedforward* and *backpropagation* algorithms are used [15]. A feedforward neural network, as a simple network, has no cycles or loops and transmits and processes data only in a forward direction from the first layer to the last. When the data is received by the input layer of this network, it is sent to the output layer via hidden layers. In each step, it processes the data based on its weights and functions, and then the obtained values are sent to the next layer [15].

A backpropagation neural network is a network that propagates the value of the difference between the output and the target, and then adjusts the edge weights. This is done by running the feedforward algorithm and processing the input data based on weights and functions to obtain an output. Next, having the error and using backpropagation, weights are adjusted in a way that the error is minimized. Backpropagation and the adjustment of weights using the mean square error and regression analysis needs to be repeated until the error

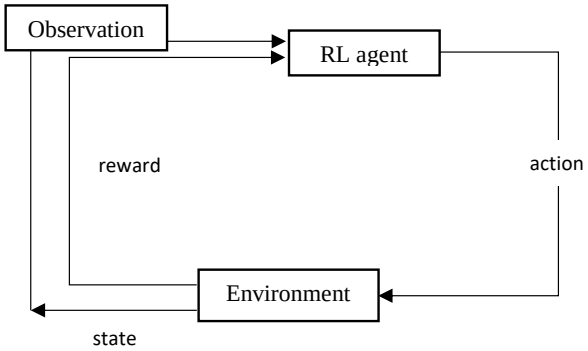


Fig. 2 Agent-environment interaction in RL

is acceptable and the performance of the system is optimized [15]. Afterwards, the training stops, and from this moment on, actual inputs are given to the input layer to predict the outputs. In other words, the network is able to model the system, predict the results, and generate a suitable output for each new input data [15].

From all the above-mentioned points, it can be concluded that ANNs can be very helpful in predicting the usage of resources, such as CPU utilization in this work. Although other less computationally complex methods have been used in previous work to predict CPU utilization, their sensitivity to data dispersion makes them prone to inaccurate predictions. By using historical data, ANNs can be used to predict CPU utilization in cloud datacenters, and to anticipate situations where host overload or underload may occur. To reach this goal, we have implemented an ANN that consists of an input layer, a hidden layer, and an output layer. Feedforward and backpropagation algorithms are used to train the network and adjust the weights.

2.2 Reinforcement Learning

Reinforcement learning (RL) algorithms can be efficiently applied to resource management in the cloud, automatic reconfiguration of shared VMs, and optimization of energy consumption, which is the subject of this work [16]. In the scope of this work, RL is adopted for VM selection to avoid system overload. RL algorithms run using an agent that aims at broadening its knowledge of its ever-changing environment by trial and error [16]. This means that should this agent choose the optimal action, it is rewarded, and if not, it is penalized, assuring that the agent will gradually gain insights into what the better option is.

RL problems are described by the Markov decision process (MDP) [17], which is a random control process for modeling sequential decisions in uncertain conditions. In MDPs, in order for the agent to choose an action, a knowledge of the current state and possible set of actions is required. Each

time, a reward is given to the agent based on the action that it has chosen. The better the action is, the greater the cumulative sum of rewards becomes, implying that the sum of received rewards is to be maximized. In an optimal situation where a complete environmental model is observable, the MDP problems can be solved by dynamic programming; otherwise, methods such as model-based RL are used for predicting the best value function or policy. RL algorithms such as *Q-learning* may be used to find optimal policies for partially observable environments.

The Q-learning algorithm belongs to the temporal difference (TD) category of RL algorithms that estimate the long-term value of an action in a particular state at a given time, denoted by a state-action pair $Q(s_t, a_t)$. Thorough knowledge of the system is not required as it estimates the long-term value of an action by exploring the environment based on the defined policy and the Bellman equation [16]. After every *state-action-reward-state* transition, Q-learning calculates the Q-value and stores it in the agent's Q-Matrix using Eq. 1.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \cdot [r_t + \gamma \cdot \max Q((s_{t+1}, a_t) - Q(s_t, a_t))] \quad (1)$$

where $Q(s, a)$ represents the expected long-term cost of selecting action a in the state s. The learning rate α determines how much information is newly acquired by the agent, meaning that should this value be zero, nothing new is learned and the prior information is still exploited, while as it gets closer to one, it is implied that future decisions will be chiefly based on the recent information. The discount factor γ holds a value between zero and one, which shows the importance of future rewards. A value close to one generates an inclination towards future rewards, whereas a value close to zero brings current rewards into focus. $maxQ(s_{t+1}, a_t)$ returns the maximum possible reward for the future state [16].

Concisely, in the standard Q-learning algorithm, the goal is to select the optimal policy leading to the maximum cumulative reward. Accordingly, Q-learning chooses the desired action from a set of possible actions based on the previously defined policies.

3 Related Work

VM consolidation plays an important role in energy consumption and resource management for datacenters, which has resulted in various research efforts in recent years. This has motivated researchers and engineers to investigate various strategies and make significant strides toward the betterment of consolidation approaches. The VM consolidation problem is completed in different phases: host overload detection, host underload detection, VM selection, and VM placement. In this section, we review some of the methods that have been put forward so far to solve this problem.

Azizi et al. [19] have introduced a heuristic-based approach for choosing a destination for VMs to enhance resource management and reduce energy consumption. This approach optimizes the number of running hosts and prioritizes them when choosing a VM destination, thus, lowering energy consumption. It also helps in the better management of resources by preventing them from being wasted and remaining idle. Khan et al. [20] have introduced a consolidation algorithm, which selects the most suitable VM, container, or application for migration. This method successfully reduces energy consumption without significant performance loss. In this method, the heterogeneity of applications is modeled by the priority of tasks, representing the workload type. The authors focused on reducing the total node costs accumulated in the inherent heterogeneous system instead of lowering the number of (physical machines) PMs. They have shown how much of an improvement the migration of applications can make when it comes to energy consumption. VM migration, on the other hand, led to better performance. The migration of applications could strike an energy-performance balance. However, The QoS, SLA, and other related metrics have not been investigated despite being an integral part of VM consolidation. Zeng et al. [21] introduced a framework for VM consolidation and employed a DRL-based algorithm and an influence coefficient to intelligently select the suitable VM for migration and a proper destination. Despite the acceptable results having been provided, metrics such as the number of VM migrations, ESV, SLATAH, and PDM have not been considered, among which the number of migrations is of paramount importance. Not only does VM migration increase energy consumption and SLA violations, but also it increases network traffic during migration. This is why in addition to the metrics mentioned above, extra attention should be paid to the number of migrations

Parvizi et al. [22] have proposed an approach for VM allocation targeting energy consumption and the number of running hosts. Their main focus is choosing the right destination for VMs so that the number of hosts and the consumption of energy can be optimized and the resources can be used more efficiently. They have treated the problem as a non-linear convex optimization and employed a non-dominated sorting genetic algorithm to find the most suitable destination for VMs. The results show that it performs better than similar approaches with regard to energy consumption. Li et al. [23] targeted energy consumption and resource utilization. For this purpose, a discrete differential evolution algorithm was presented to select the best host for the placement of migrated VMs. The results show that it positively affected energy consumption and QoS by minimizing SLA violations. The only weakness, however, was the lack of attention to the number of migrations. Optimizing this number plays a huge role in real-life datacenters in addition to the management of energy consumption and host overload.

Khan [24] has introduced a normalization-based method for VM consolidation to reduce energy consumption, SLA violations, and the number of VM migrations. To detect overloaded and underloaded hosts in a cloud environment, the parameters related to VMs and hosts were used. It should be noted that for host overload detection, static thresholds were used. Con-

sidering the dynamic characteristics of real workloads, it is required to use dynamic thresholds as well. The performance of this method, however, was acceptable in reducing energy consumption and SLA violations. Ranjbari et al. [25] introduced an automata-based learning algorithm to reduce energy consumption and SLA violations in the area of VM consolidation. This algorithm utilizes learning automata to predict CPU utilization and to identify overloaded hosts. Although it has been done successfully, the reduction of energy consumption and the number of VM migrations still need more investigation. Beloglazov et al. [26] have introduced five methods for host overload detection. One method is the static threshold (THR), which, as the name implies, uses a static threshold to decide whether or not the host is overloaded. This means that THR lacks the ability to adapt to varying workloads. The other four methods adjust the threshold based on the historical data, which helps in estimating when migrations might happen under different load conditions. They are the interquartile range (IQR), the median absolute deviation (MAD), the local regression (LR), and the local robust regression (LRR). The authors of [26] have also introduced three VM selection methods, including minimum migration time (MMT), maximum correlation (MC), which selects the VM with the highest correlation of CPU utilization with other VMs, and random choice (RC), which randomly chooses VMs. It should be noted that these methods can be used independently or in combination with other approaches. We have used the combinations of the above-mentioned methods (IQR-MC, IQR-MMT, LR-MC, LR-MMT, LRR-MC, LRR-MMT, MAD-MC, MAD-MMT, THR-MC, THR-MMT) to evaluate the approach proposed in this paper.

Monil et al. [27] have proposed an algorithm based on fuzzy logic for the selection of a suitable VM for migration. The overload detection was also done based on the CPU usage standard deviation. This method could save energy and achieve favorable results regarding QoS improvement and SLA violations. However, using the mean value is a pitfall in this method as it is extensively influenced by minimum and maximum values. Han et al. [28] used the Markov decision process (MDP) to model the problem of VM management, along with the value iteration algorithm [29], which tries to choose the best policy for VM management in every step. However, the large state space of the problem required them to put forward an approximate dynamic VM management method based on MDP to overcome the problem of dimensionality. Rasouli et al. [30] have introduced a learning automata-based approach to reduce energy consumption in the process of VM migration. This approach uses CPU utilization as the sole parameter for choosing destination hosts with no knowledge of the applications running on the cloud. The results show that it significantly reduces energy consumption and satisfies QoS. Wu et al. [31] aimed at optimizing energy cost with minimum migration cost. They proposed an improved grouping genetic algorithm (IGGA) to calculate the maximum value of the consolidation score by dynamic VM consolidation. In a consolidation score function, the two conflicting objectives are normalized and summed

up with different preference weights. This method successfully reduced energy consumption but was not as successful in lowering SLA violations.

Hallawi et al. [32] proposed a genetic-based approach for VM consolidation and resource management. The approach aims at the reduction of running hosts and the optimization of resource usage. For this purpose, they developed two genetic algorithms, namely COFFGA and CONFGA, to help with the efficient use of resources. This is done by obtaining a proper order of VMs as the solution evolves. Despite these improvements, the adoption of first-fit decreasing (FFD) for VM placement adversely affects QoS and the availability of reserved resources that increases SLA violations. Monil et al. [33] worked on a method that provides a tradeoff between QoS and energy consumption in VM consolidation. This method uses best-fit descending bin packing to reduce the number of hosts and consolidate VMs efficiently. The authors decomposed the dynamic VM consolidation problem into multiple sub-problems and proposed an approach that states that an already shutdown host cannot be activated unless there is no better option. To achieve this goal, an optimization phase was introduced.

Telenyk et al. [34] proposed a method based on simulated annealing to optimize energy consumption and improve QoS. Although this approach successfully met those expectations, it failed in handling the host re-overload as it does not take the stochastic resource demands into account. Li et al. [35] adopted the bee colony optimization to decrease both the energy consumption and the number of VM migrations. They also detect host overload and underload, however, the convergence speed is still comparatively low. Lu et al. [36] used proactive resource management to present new host detection techniques that operate based on forecasting exchange rates using an exponentially weighted moving average. Their approach could enhance QoS, reduce energy consumption, and help in the process of host overload detection. Tarahomi et al. [37] aimed at reducing energy consumption using an evolutionary algorithm for VM allocation. They have designed a micro-genetic algorithm to find a suitable destination for VMs. The fitness value in this algorithm is calculated based on the total power consumption obtained from CPU utilization. This algorithm is applied to overloaded or underloaded hosts whenever a migration is going to take place. It searches the state space effectively and its fast convergence makes it efficient for selecting a suitable VM destination while reducing energy consumption and SLA violations.

Liu et al. [38] introduced a dynamic virtual machine consolidation method that migrates VMs before hosts overload to save energy and guarantee QoS. They address the four main problems of VM consolidation (overload detection, underload detection, VM selection, destination host selection) using an autoregressive integrated moving average (ARIMA) model. This model obtains a stationary time series from an original time series to predict the future host CPU utilization. An adaptive reserved resources model was also proposed to prevent host re-overload. Aslam et al. [39] succeeded at reducing energy consumption by using neural networks, along with feedforward and backpropagation algorithms for the selection of the proper VM for migration. This approach

utilizes factors such as the amount of RAM, CPU usage standard deviation, and correlation of each VM for prediction and VM selection, but it neglects the overall status of servers after the migration. Basu et al. [40] introduced Megh, an online RL-based algorithm with reduced computational overhead throughout the live migration of VMs. This method learns from the dynamic real workload and models the problem of energy consumption and resource management in VM migration as an MDP and solves it using a functional approximation scheme. The complexity of the state space of this problem was also reduced to a polynomial dimensional space with a sparse basis using a projection method.

4 The proposed approach

In this paper, the infrastructure-as-a-service (IaaS) type of cloud systems is considered. Cloud service providers offer computing, storage, and network resources on demand. We assume that the system consists of M heterogeneous hosts. The performance of these hosts can be described by the CPU capacity, defined in millions of instructions per second (MIPS), the amount of RAM, and the network bandwidth. In this system model, each user is assigned a VM chosen from the N existing VMs. In order to achieve versatility and adaptability in varying conditions, no assumptions are made regarding the applications and machines' workload, while CPU utilization is considered a prime factor for defining workload.

As mentioned earlier, for the sake of clarity, we break down the VM consolidation problem into four sub-problems: (1) detecting any cases of host overload, (2) choosing a proper VM for migration, (3) detecting any cases of host underload, and finally, (4) choosing a suitable destination host on which the VM can be placed. This method of breaking down the VM consolidation problem into multiple parts has been adopted by researchers [20]. The herein proposed approach addresses the first three sub-problems in three separate parts, as it is illustrated in Fig. 3, and described in the following.

- At the left-hand side of Fig. 3, an ANN is used for host overload and underload detection to reduce energy consumption and SLA violations by predicting the host CPU utilization and migrating VMs. Host overload and underload detection algorithms are presented in Sections 4.1 and 4.3.
- At the right-hand side of Fig. 3, RL is used for the smart selection of VMs on overloaded hosts, minimization of the number of unnecessary migrations, and reduction of SLA violations. The proposed RL is described in Section 4.2.
- At the bottom of Fig. 3, after detecting overloaded hosts and selecting VMs for migration, they are migrated to normal hosts. The underloaded hosts can also be shut down after having their VMs migrated to other hosts with enough capacity. This way, resources are managed more efficiently, energy consumption is reduced, and SLAs are prevented from being violated. This algorithm is presented in Section 4.3.

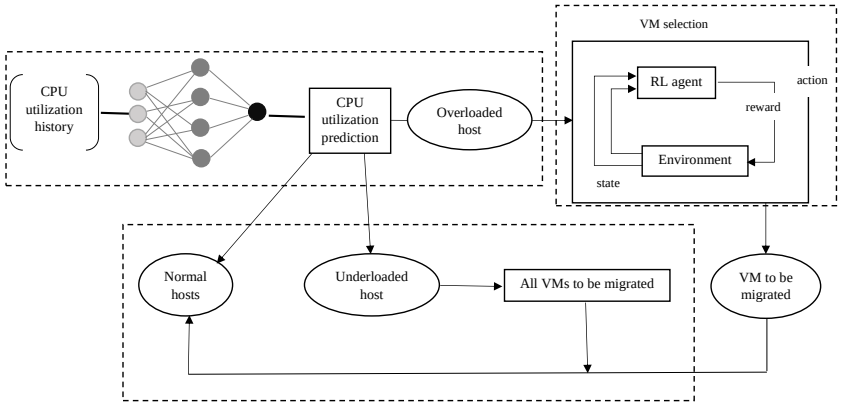


Fig. 3 The entire process of the proposed approach

It should be noted that the last problem of VM consolidation (selection of a suitable destination host) is done using a modified version of the best fit decreasing (BFD) algorithm introduced in [13]. In this algorithm, VMs are sorted in decreasing order of their current CPU utilization, and using the heterogeneity of resources, each of them is allocated to a host that appears to result in the minimum rise in energy consumption and is the most power-efficient.

4.1 Proposed ANN for host overload detection

To take the first step towards VM consolidation, as it is shown in Fig 3, an ANN is applied to predict host utilization and avoid host overload. Despite all the improvements in this area, there are still some hidden parameters that can impact the performance of servers, such as the details involved in hardware design. Due to the abundance of these factors, using an analytical model is almost impossible and inefficient as it can become useless with the advent of next-generation processors with system architectures that differ from the old ones. This makes it more suitable to choose a more flexible and time-saving solution such as ANNs, which can provide a satisfying model once you have the suitable dataset for training. In the first step, the feed forward algorithm is used to propagate data from the input to the output layer. Then the backpropagation algorithm, iterating backward from the last to the first layer, is used for training the network until weights are efficiently adjusted and minimum error is produced.

The inputs of the proposed ANN are gathered from the last 10 iterations of hosts' past CPU utilization measured at regular intervals. As it is shown in Algorithm 1, for any host j, the historical CPU utilization data set $\{H^{u_j}(n-9), H^{u_j}(n-8), H^{u_j}(n-7), ..., H^{u_j}(n)\}$ is given to the network as the input set. Since $H^{u_j}(n)$ denotes the amount of CPU utilization for host j at each iteration, a well-trained neural network is able to make use of this historical

data to predict future CPU utilization. Next, overloaded hosts are detected, which is done by checking whether the condition $H^{u_j}(n+1) > 1$ is satisfied or not. If this is the case, the host is categorized as overloaded.

Algorithm 1 Detection of overloaded hosts

Input: HostList
Output: OverloadedHostList

```
for each host in Hostlist do
    input_neural ← UtilizationDataSet
    output_neural ← predictedCPU (UtilizationDataSet)
    PredictUtil(host) ← output_neural
    if PredictUtil(host) > 1 then
        OverloadedHostList.add(host)
    end if
end for
return OverloadedHostList
```

4.2 Proposed RL for VM selection

In order to reduce the load of the hosts detected in the previous stage, some VMs should be migrated from overloaded hosts to appropriate destinations. This may cause performance degradation and be reflected in the QoS, so the meticulous selection of the proper VM is key when handling SLA violations. The ability of RL agents to learn by observing the environment and adapting to its dynamic nature makes them a suitable choice to overcome these challenges. RL techniques have also been used in public clouds that provide IaaS for the auto-configuration and the migration of VMs [40–43]. In contrast to the previous use cases of RL techniques that mostly revolved around the reduction of energy consumption, the proposed approach employs RL to reduce the number of migrations and improve the QoS. Research has shown that preventing frequent and unnecessary migrations by selecting the right VM can significantly boost resource management and reduce energy consumption and SLA violations. As it is shown in Fig 3, the suitable VM for migration is selected by the RL algorithm after the detection of overloaded hosts. The details of the proposed RL algorithm are presented in the following subsection.

4.2.1 State and action spaces

The state space is defined as the percentage of the CPU utilized by overloaded hosts. On every overloaded host, there is a set of VMs running and using a specific proportion of the CPU, meaning that the total CPU utilization at any given time is equal to the sum of each individual VMs' utilization. Based on this, the state space ranges from 0 to 100, or in other words: $s_t \in S = \{0 : 100\}$.

The action space of Algorithm 2 is equal to the set of VMs running on the host. The agent chooses an action from the set and receives reward feedback

from the environment to evaluate the choice. The environment also enters a new state after the selection of the VM. As the interaction of the agent and the environment continues, the agent learns what actions are better to take. It chooses an action based on the greedy-ϵ policy at each step, which might end up being an exploration instead of exploitation. The agent uses this knowledge to update Q-values. The Q-matrix of the RL algorithm is updated by the Q-value obtained from the Bellman equation for each state-action pair [16].

Algorithm 2 VM Selection

Input: OverloadedHost h
Output: VMtoMigrate , VM_m

Take action a_t : VM_m ← Choose VM from set of VMList(h) using policy
Migrate selected VM
Observe state s_{t+1}
Reward ← calculate reward using Eq. 3
$Q(s_t, a_t)$ ← calculate $Q(s_t, a_t)$ using the Bellman equation
update Q-matrix
return VM_m

4.2.2 Reward function

An RL agent tries to achieve the highest possible reward during the process of mapping actions to states, which makes frequent interactions with the environment necessary. When the agent at state s_t receives information and chooses action a_t, a reward commensurate to the quality of that action is assigned to it through the reward function $R(s_t, a_t)$. This way, the agent learns how smart its move has been and tries to improve it in future iterations. This reward might fluctuate, but as the process continues, reward values converge to the expected value, which leads to Q-value converging over time. The reward function is defined according to the CPU usage standard deviation factor.

One way to achieve energy efficiency and reduce traffic load is migration control. Using migration control methods, we are able to identify the VM that regularly consumes resources and take control over its migrations [44]. One of the root causes of host overload is the CPU usage of a fluctuating VM. When VM CPU requests change erratically, the host might not be capable of provisioning resources when they are required, which leads to SLA violations. In other words, the less deviated VM utilization values are from the standard, the less likely it is for the host to become overloaded. This concept is used to select the suitable VM and define the reward function. Since host utilization can be calculated using the utilization of running VMs, similarly, the standard deviation of the CPU usage of these VMs (within N time frames) can be used to calculate that of the associated host. Eq. 2 shows how the average of the sum of M VMs' standard deviations can be used to calculate this value.

Table 1 The configuration of hosts

Hosts	CPU type	Frequency (GHz)	Cores	RAM (GB)
HP ProLiant G4	Intel Xeon 3040	1.86	2	4
HP ProLiant G5	Intel Xeon 3075	2.86	2	4

$$Stdev_{host} = \sqrt{\frac{\sum_{i=1}^{N}(CPU^{i}_{VM_1} - CPU^{avg}_{VM_1})^2 + \ldots + \sum_{i=1}^{N}(CPU^{i}_{VM_M} - CPU^{avg}_{VM_M})^2}{M*N}} \quad (2)$$

In Eq. 2, $CPU^{i}_{VM_1}$ is the *ith* CPU usage of VM_1 and $CPU^{avg}_{VM_1}$ is the average CPU usage over the course of N iterations. Then, in order for these policies to be reflected in the RL algorithm and prevent SLA violations, the reward function is written as Eq. 3.

$$Reward = 1 - Stdev_{host} \quad (3)$$

4.2.3 VM selection procedure

As specified in Algorithm 2, the RL agent calculates the states and set of actions for the overloaded host identified by the ANN proposed in Section 4.1. VMs suitable for migration are detected and selected by the agent based on the defined policy and Q-values for each action-state pair in the Q-matrix. Once the VM is selected, it is migrated to a proper host, but the process does not finish before the reward is assigned in proportion to the quality of the action. The standard deviation-based reward is calculated and the Q-value for each state-action pair updates the Q-matrix. Unless the host workload is balanced, the RL agent continues the process of VM selection based on the feedback given by the reward function.

4.3 Proposed ANN for host underload detection

The detection of underloaded hosts is also a significant part of VM consolidation. As it is illustrated in Fig 3, during this stage, all the VMs running on the underloaded host are migrated to other suitable hosts so that the underloaded host can be shut down. This is a fruitful mechanism for energy efficiency and efficient resource management. Once all the required migrations have taken place, CPU utilization values are predicted for hosts that were not categorized as either overloaded or allocated (hosts chosen as VM destinations). As it is explained in Algorithm 3, the host with the least predicted CPU utilization value of below 30% is defined as underloaded.

Algorithm 3 Detection of underloaded host

Input: HostList
Output: Underloadedhost

```
HostList.remove(OverloadedHostList)
HostList.remove(AllocatedHostlist)
HostList.remove(SwitchedOffHostlist)
minhostUtil = 1
Underloadedhost = NULL
for each host in Hostlist do
    input_neural ← UtilizationDataSet
    output_neural ← predictedCPU(UtilizationDataSet)
    PredictUtil(host) ← output_neural
    if PredictUtil(host) < minhostUtil and 30% then
        minhostUtil ← PredictCpuUtil(host)
        Underloadhost ← host
    end if
end for
return Underloadedhost
```

Table 2 Power consumption of servers at different load levels in Watts

Server \ Utilization	sleep	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
HP ProLiant G4	10	86	89.4	92.6	96	99.5	102	106	108	112	114	117
HP ProLiant G5	10	93.7	97	101	105	110	116	121	125	129	133	135

5 Numerical Results

This section compares the proposed approach to other related methods and evaluates its performance and efficiency based on different experiments carried out with the CloudSim framework [45]. The data and setup for our experiment are similar to what researchers working in this area had previously used. In the following, the experimental setup and performance metrics are introduced, and then numerical results are provided.

5.1 Experiment Setup

CloudSim is a simulation toolkit designed to offer a platform for the test and evaluation of new policies and strategies before being applied to real datacenters [45]. The heterogeneous datacenter we simulated in our experiments encompasses 400 HP ProLiant ML110 G4 servers and 400 HP ProLiant ML110 G5 servers. The configuration of hosts used in these experiments is provided in Table 1. Moreover, Table 2 and Table 3 represent the power consumption of servers for different percentages of utilization, and the characteristics of VMs, respectively.

The data used for the evaluation of the proposed approach is the real workload collected from the CoMon project [46]. These data are included in PlanetLab files in the CloudSim framework, which includes CPU utilization

Table 3 VM types

VM type	CPU (MIPS)	RAM (GB)
High CPU medium instance	2500	0.85
Large instance	2000	1.70
Small instance	1000	1.70
Micro instance	500	0.61

Table 4 PlanetLab trace data

Date	VMs
2011/03/03	1052
2011/03/06	898
2011/03/09	1061
2011/03/22	1516
2011/03/25	1078
2011/04/03	1463
2011/04/09	1358
2011/04/11	1233
2011/04/12	1054
2011/04/20	1033

of VMs running on thousands of servers located in about 500 different places around the globe. This project measures CPU utilization every five minutes within 24 hours. As shown in Table 4, data for 10 different days are considered.

The proposed ANN is trained using the feedforward and backpropagation algorithms. The network is trained using the k-fold cross-validation method with k equal to 10. This method divides the dataset into k chunks, $k-1$ of which are used for training, and one is used for validation. The dataset used in this work reflects the workload of a real system. This workload includes the CPU utilization of VMs that are highly independent and different from each other. It has also been collected within 10 random days [45].

A three-layer feedforward network is trained, which includes an input layer of ten nodes, a hidden layer of six nodes, and an output layer of one node. The transfer functions for the hidden units and output units are a sigmoid and a linear function, respectively. These units are fully connected, meaning that every unit in the input layer is connected to every other unit in the hidden layer and every unit in the hidden layer is connected to every other unit in the output layer. Weights are initialized randomly and the learning rate is set to 0.1. As was mentioned in Section 4, the input data (host CPU utilization) is updated by weights, propagated to the hidden layer, and then, to the output layer. In the proposed approach, to minimize the error rate, the backpropagation algorithm is used for training. In our experiments, the epoch value, denoting the number of times the algorithm sees the entire training

dataset, is set to 50. This value is incremented every time a forward and a backward pass takes place and a cycle is completed.

In the process of RL training, the agent learns to choose the suitable VM by following the defined policy, meaning that considering the current state of the system, it chooses the action resulting in the greatest Q-value. For the RL algorithm to converge toward the desired policy, RL parameters should be adjusted considering the problem requirements. In our experiments, parameters, α, γ, and ϵ are set to 0.5, 0.7, and 0.01, respectively.

5.2 Performance metrics

The main goal of the approach presented in this paper is to reduce both energy consumption and SLA violations to keep QoS at an acceptable level. For this purpose, the six below-mentioned metrics are adopted to evaluate the proposed approach.

– **Energy Consumption (EC)**
It shows the amount of energy consumed by physical hosts in the datacenter. It has a direct relation with the power consumption of CPU, memory, storage, and VM migrations [47,48]. Research has shown that the power consumption of physical hosts can be defined as a linear function of CPU utilization, as shown in Eq. 4 [49],

$$P(h) = K * P_{max} + (1 - K) * P_{max} * h^u \tag{4}$$

where P_{max} shows the maximum power required when the host's CPU is fully utilized (100%), the constant K, set to 0.7 based on previous research, is the percentage of power consumption for an idle host, and h^u denotes the current host CPU utilization [50]. The CPU utilization variations over time indicate that the CPU utilization of the host can be described as a function of time, which helps us to define the host's total energy consumption by Eq. 5.

$$EC = \int_{t_0}^{t_1} P(h^u(t))dt \tag{5}$$

– **Total Number of VM Migrations (VMMigNo)**
Live migration techniques aim to transfer a VM between physical hosts in a shorter amount of time without experiencing any suspension [51,52]. Despite this merit, live migration negatively affects the performance of VMs, resulting in average performance degradation of 10% as previous research has shown [53]. The migration time and the performance degradation due to migration can be approximately expressed by Eq. 6 and Eq. 7, respectively [26].

$$T_{m_j} = \frac{M_j}{B_j} \tag{6}$$

$$U_{d_j} = 0.1 * \int_{t_0}^{t_0+T_j^m} u_j(t)dt \tag{7}$$

In Eq. 6, M_j denotes the amount of RAM used by the VM and B_j represents the network bandwidth. In Eq. 7, U_{d_j} is the total performance degradation for VM j. t_0 and T_j^m represent the starting and the completion times of the migration respectively, and u_j shows the amount of CPU utilization by VM j.
From these models, it can be easily deduced that VM migration is a contributing factor in performance degradation, which gives rise to SLA violations; Thus, the less frequent VM migrations are, the fewer SLA violations occur.

– **SLA violation Time per Active Host (SLATAH)**
This metric is the percentage of time during which active hosts undergo a period of overload due to 100% CPU utilization. It is known that it significantly limits the performance of the system when applications running on the host fully utilize the CPU. Consequently, VMs will not be able to fulfill the requirements of clients [54]. This means that the application will not get the required performance from the host which results in SLA violations. This value is calculated using Eq. 8 [26],

$$SLATAH = \frac{1}{N}\sum_{i=1}^{N}\frac{T_{s_i}}{T_{a_i}} \tag{8}$$

where N is the number of physical hosts. Moreover, T_{s_i} and T_{a_i} represent the total time during which the utilization of host i is 100% and the total time during which host i is active, respectively.

– **Performance Degradation due to Migrations (PDM)**
Overall performance degradation caused by migration can be calculated by Eq. 9,

$$PDM = \frac{1}{M}\sum_{j=1}^{M}\frac{C_{d_j}}{C_{r_j}} \tag{9}$$

where M is the number of VMs, C_{r_j} represents the total CPU capacity of VM j requested during the time it is active, and C_{d_j} is an estimation of performance degradation of VM j, happening due to VM migrations. This estimation is based on the performance of the VM before and during migrations. Similar to the previous metric, this adversely affects the ability of the system to provide applications with the performance they require, which creates SLA violations.

– **SLA Violations Cost (SLAV)**
Since the level of service expected by clients from a provider is of paramount importance, various metrics such as the minimum CPU capacity or the maximum response time of the system are agreed upon in contracts called SLA. These metrics might differ across various applications. This is why it is important to define a workload-independent metric when evaluating the quality of services. In this work, we assume that SLA has been met only if, at any time interval, the performance requirements of the applications are fully (100%) met by the host. For measuring SLA violations in an IaaS

environment, the above-mentioned metrics (SLATAH and PDM) can be used. Each of these two metrics independently measures SLA violations; however, in order to intelligently exploit the features both of these metrics offer, the *SLAV* metric is derived from their combination.

$$SLAV = SLATAH * PDM \tag{10}$$

As Eq. 10 suggests, SLAV takes into account both overload-related and migration-related performance degradation.

– **Energy Consumption and SLAV (ESV)**
As Eq. 11 suggests, ESV is calculated as the product of EC and SLAV metrics. Based on this combination, the smaller this value is, the more efficient the approach will be.

$$ESV = EC * SLAV \tag{11}$$

5.3 Comparison and performance analysis

To evaluate the interest of the proposed approach, results are compared with the related state-of-the-art methods, including DTHMF [33], EQVC [38], and MMSD-FS [27], in addition to 10 baselines named IQR-MC, IQR-MMT, LR-MC, LR-MMT, LRR-MC, LRR-MMT, MAD-MC, MAD-MMT, THR-MC, AND THR-MMT. Table 5 provides results obtained for the metrics mentioned in Section 5.2. These are the average values obtained from running each algorithm using PlanetLab's workload. Results show that the proposed approach provides a successful energy-SLA balance by predicting hosts' future status and improving the VM selection process. This means that it outperforms baselines, and achieves better results than state-of-the-art methods in most of the metrics, which makes it more efficient overall.

In Table 5, it can be observed that the proposed approach shows improvements when compared to all the other algorithms. Regarding PDM, our approach successfully reduces the number of VM migrations that contribute to a larger PDM, by requesting and occupying the CPU capacity more productively. The prediction strategy and overload detection using ANN not only lowers the number of overloaded hosts but also optimizes the number of migrations, due to host overload, thus improving the PDM value. From Fig. 4 it can be concluded that, on average, our approach reduces PDM by 88% in comparison to the baselines. The comparison with the state-of-the-art algorithms also shows a 63% decrease in PDM.

The SLATAH metric shows how host overload can affect the QoS. Fig. 5 demonstrates that the proposed approach outperforms other algorithms by minimizing SLATAH. From Fig. 5, it is also noticed that an state-of-the-art method such as DTHMF, along with baselines such as LR and LRR, account for high values of SLATAH under varying workloads. However, from the perspective of energy consumption, Table 5 shows that these methods are efficient since the strategy they have adopted consolidates VMs to a great degree but

Table 5 Performance evaluation using PlanetLab with six metrics

Strategy \ Metric	EC (KWh)	VMMigNo	SLAV (10^{-5})	SLATAH (%)	PDM (%)	ESV (10^{-2})
Proposed Approach	123.10	2502	0.2530	2.46	0.010	0.031
EQVC	111.52	6552	0.638	2.55	0.020	0.71
DTHMF	146.23	13609	3.600	9.0	0.040	0.526
MMSD-FS	136.50	7943	0.500	2.50	0.020	0.068
LR-MMT	161.87	27632	4.970	6.231	0.080	0.804
LR-MC	148.50	23929	7.608	7.484	0.102	1.129
THR-MMT	188.50	26602	3.368	5.038	0.068	0.634
THR-MC	179.37	23956	6.995	6.911	0.101	1.254
MAD-MMT	183.49	26301	3.348	5.284	0.063	0.614
MAD- MC	173.82	23418	7.111	7.053	0.100	1.236
IQR-MMT	187.50	26497	3.288	5.023	0.060	0.616
IQR-MC	177.70	23393	6.805	6.914	0.098	1.209
LRR-MMT	161.87	27632	4.970	6.231	0.080	0.804
LRR-MC	148.50	23929	7.608	7.484	0.102	1.129

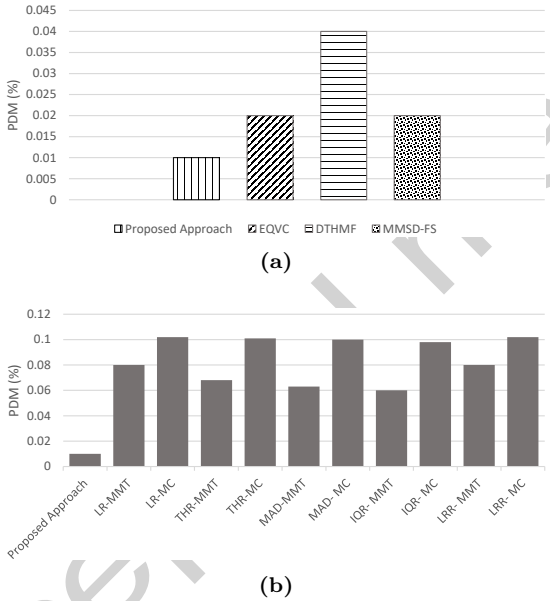


Fig. 4 Performance degradation due to migrations (PDM) calculated for the (a) state-of-the-art, and (b) baseline algorithms

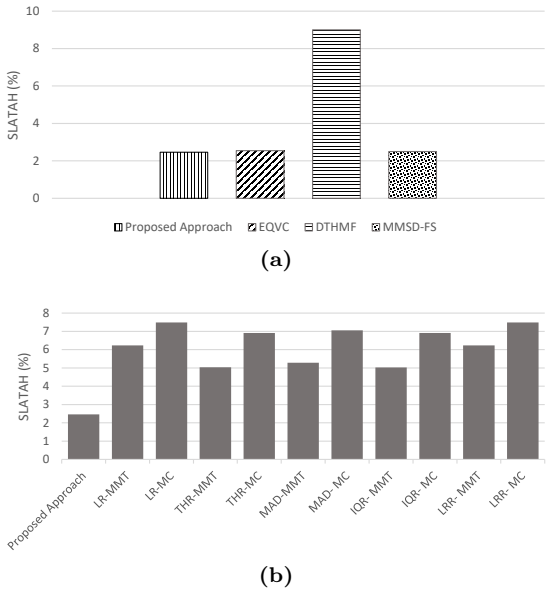


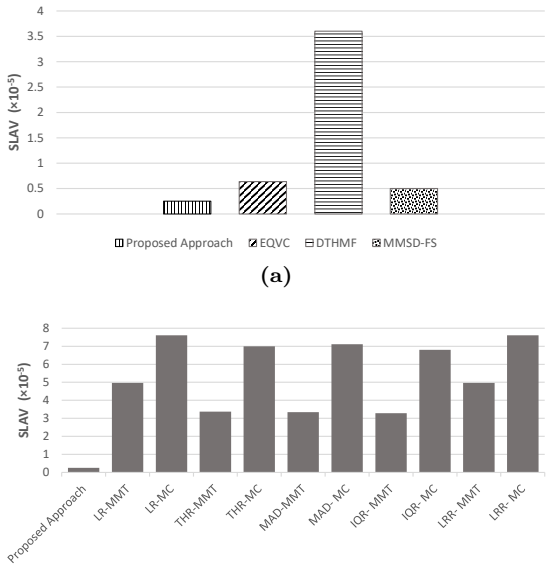
Fig. 5 SLA violation time per active host (SLATAH) calculated for the (a) state-of-the-art, and (b) baseline algorithms

at the cost of increasing the probability of host overload. MMT performs better than MC in this regard as it reduces the VM migration time and CPU utilization throughout the migration process, which has favorable effects on host overload. Despite this, the ability of the proposed approach to predict host overload in the upcoming step makes it the most efficient by reducing SLATAH by around 58% on average, for both the baselines and the state-of-the-art.

The efficiency of the proposed approach in avoiding host overload promotes the availability of resources, which consequently, prevents SLA violations. According to Fig. 6, this improvement is 95% and 83% for the baselines and the state-of-the-art algorithms, respectively. It should also be noted that since the SLAV metric is closely linked to QoS, customers can experience a better QoS with the proposed approach.

Based on the results in Fig. 7, the proposed approach is responsible for the fewest number of migrations by holding the value of 2502, while the average number of migrations for the baselines and state-of-the-art are 26567 and 9368 respectively. It shows that the proposed approach significantly outperforms other approaches as the number of migrations resulting from it is much lower than that of other approaches shown in Table. 5. This means that when the proposed approach is employed, the VMMigNo can result in lower energy consumption by declining by 90% and 73% for the baselines and state-of-the-art algorithms, respectively.

The smart detection of overloaded and underloaded hosts prevents frequent host shutdowns and plays an important role in enhancing the process



(a)

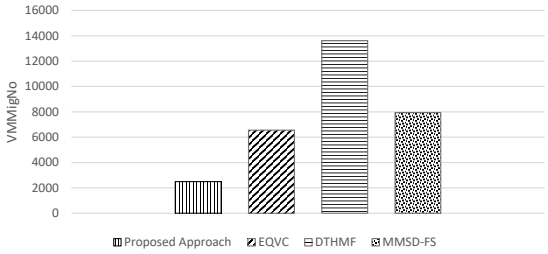
(b)

Fig. 6 SLA violations cost (SLAV) calculated for the (a) state-of-the-art, and (b) baseline algorithms

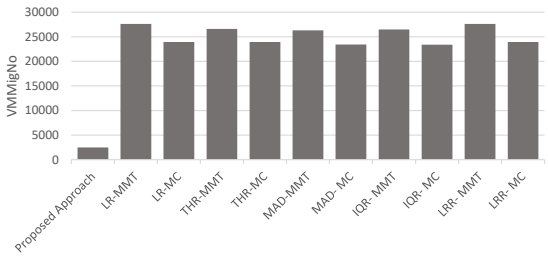
by minimizing the number of unnecessary migrations. This way, performance degradation can be controlled, meaning that SLA violations can be reduced and energy can be saved.

Fig. 8(a) shows that the proposed approach improves energy consumption except for EQVC. Although this algorithm consumes less energy, it does not perform as efficiently as the proposed approach when other metrics are also taken into account. Our approach is 29% more energy-efficient than baselines. Although less perceptible, it improves EC by 3% for the state-of-the-art algorithms. This is even more pronounced in the results of IQR-MMT and THR-MMT, which rank first and second for the highest energy consumption due to weak predictions. As it is presented in Fig. 8(b), costs are higher for MMT as this algorithm solely takes the migration time into account. This leads to the selection of unsuitable VMs and gives rise to energy consumption. MC is another VM selection approach, which performs better than MMT. The huge number of migrations, however, hinders its ability to reduce energy consumption, whereas the proposed approach successfully overcomes this problem.

The last metric considered herein is ESV, which gives insights into both EC and SLAV parameters. As it can be seen in Fig. 9, the proposed approach could once again surpass the other approaches, meaning that it can guarantee high QoS while saving energy. It also goes further and lowers operational costs, which makes it a win-win strategy for both customers and providers. Neither of the DTHMF and MMSD-FS methods, despite being among the recent research works, could perform better than the proposed approach in

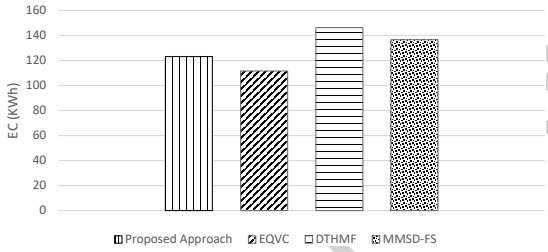


(a)



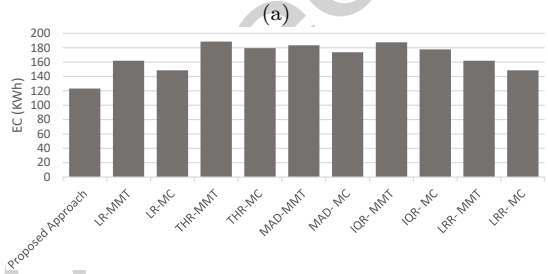
(b)

Fig. 7 Total number of VM migrations (VMMigNo) calculated for the (a) state-of-the-art, and (b) baseline algorithms



(a)

(a)



(b)

Fig. 8 Energy consumption (EC) calculated for the (a) state-of-the-art, and (b) baseline algorithms

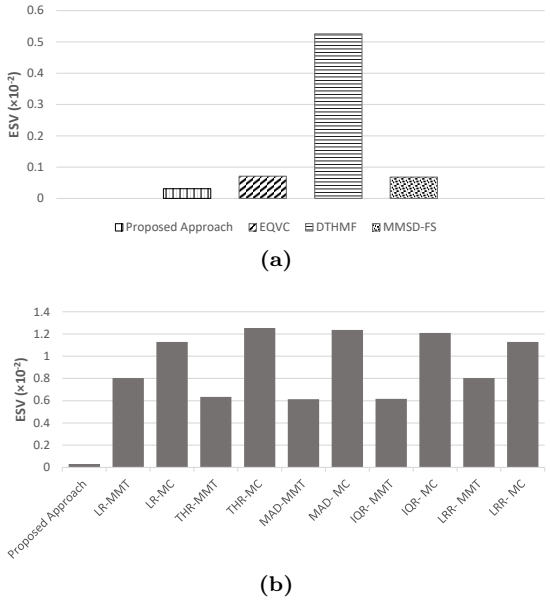


Fig. 9 Energy consumption and SLAV (ESV) calculated for the (a) state-of-the-art, and (b) baseline algorithms

ESV since the number of migrations is still higher. The figures show that our approach is 90% better than the state-of-the-art algorithms, and compared to the baselines, the ESV improved drastically by 96%. In addition to the above methods, we have compared the proposed approach to the state of the art, such as EPC [20], NVMC [24], and the proposed algorithm in [37]. The results indicate that compared to EPC, the energy consumption is reduced by 32% when the proposed approach is applied. This value is 22% when it is compared to the method in [37]. Next to NVMC, the SLA violations and the number of migrations are also reduced by 47% and 14%, respectively.

Experimental results show that using ANNs to predict the CPU utilization of hosts helps us achieve better results than algorithms that predict the CPU utilization of only one host, such as LR and LRR, or state-of-the-art algorithms such as MMSD. Using RL algorithms for VM selection could effectively lower the number of migrations compared to MMT, EQVC, and DTHMF and prevent host overload. This allows the proposed approach to overtake others in saving energy and guaranteeing QoS. The proposed approach proved to have the potential for helping providers use resources more intelligently, not only to lower datacenter costs but also to provide customers with high QoS, and pave the way for the development of cloud-based services.

6 Conclusions and future work

The dynamic nature of datacenter workloads has intensified the role of proper resource management and energy efficiency. Thus, various approaches have been put forward, among which we can highlight VM live migration from underloaded and overloaded hosts. These approaches, however, face challenges. In this paper, we proposed a new approach using ANNs and the RL algorithm. With ANNs and the completion of learning using the training set, we predict the future CPU utilization of hosts. Subsequently, the RL algorithm is used to select the suitable VM for migration. After the process of VM migration is over, the proposed approach detects the underloaded host. This way, not only does our approach detect host overload, but also intelligently selects VMs for migrations and identifies underloaded hosts that can be shut down. We have used Planetlab's real workload to evaluate the proposed approach and compare it to other algorithms. The results show that the proposed approach effectively reduces energy consumption, improves QoS under varying workloads, optimizes the number of migrations, and reduces SLA violations. This leads to a more stable system with fewer hosts running, thus reducing the energy consumption of the whole datacenter.

VM consolidation consists of four parts, three of which have been addressed in this paper. For future work, we will look into the fourth part, the selection of the most suitable destination for a VM. Although CPU is one the most significant factors affecting the energy consumption of hosts, we want to consider other contributing factors including memory, disk storage, and workload. This way, we can further improve upon our current approach by optimizing the number of necessary migrations and preventing host overload.

Author contributions. Mahshid Rezakhani: Conceptualization, methodology, validation, formal analysis, investigation, data curation, writing the original draft. **Nazanin Sarrafzadeh-Ghadimi:** Conceptualization, writing, review, and editing. **Reza Entezari-Maleki:** Conceptualization, methodology, review, and editing, supervision. **Leonel Sousa:** Conceptualization, methodology, review, and editing. **Ali Movaghar:** Conceptualization and supervision.

All authors reviewed the manuscript.

Funding. No funding was received for this research.

Data availability. No data is available for this research.

Declarations

Conflict of interest. The authors declare that they have no conflict of interest.

Ethical statement. This research is authors' original work, and it has not received prior publication and is not under consideration for publication elsewhere.

Informed consent. Informed consent was obtained from all individual participants included in the study.

References

1. M. N. O. Sadiku, S. M. Musa, , O. D. Momoh, Cloud computing: Opportunities and challenges, IEEE potentials 33 (1) (Jan.-Feb., 2014) 34–36.
2. R. Entezari-Maleki, L. Sousa, A. Movaghar, Performance and power modeling and evaluation of virtualized servers in IaaS clouds, Information Sciences 394–395 (2017) 106–122.
3. S. Ilager, K. Ramamohanarao, R. Buyya, ETAS: Energy and thermal-aware dynamic virtual machine consolidation in cloud data center with proactive hotspot mitigation, Concurrency and Computation: Practice and Experience 31 (17) (2019) e5221.
4. E. Ataie, R. Entezari-Maleki, E. Etesami, B. Egger, D. Ardagna, A. Movaghar, Power-aware performance analysis of self-adaptive resource management in IaaS clouds, Future Generation Computer Systems 86 (2018) 134–144.
5. A. H. T. Dias, L. H. A. Correia, N. Malheiros, A systematic literature review on virtual machine consolidation, ACM Computing Surveys 54 (8) (2022) 176:1–176:38.
6. P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, , A. Warfield, Xen and the art of virtualization, ACM SIGOPS Operating Systems Review 37 (5) (2003) 164–177.
7. P. Li, S. Guo, T. Miyazaki, X. Liao, H. Jin, A. Y. Zomaya, , K. Wang, Traffic-aware geo-distributed big data analytics with predictable job completion time, IEEE Transactions on Parallel and Distributed Systems 28 (6) (Jun. 2017) 1785–1796.
8. A. Iosup, S. Ostermann, M. N. Yigitbasi, R. Prodan, T. Fahringer, , D. Epema, Performance analysis of cloud computing services for many-tasks scientific computing, IEEE Transactions on Parallel and Distributed Systems 22 (6) (Jun. 2011) 931–945.
9. G. Taheri, A. Khonsari, R. Entezari-Maleki, M. Baharloo, L. Sousa, Temperature-aware dynamic voltage and frequency scaling enabled MPSoC modeling using stochastic activity networks, Microprocessors and Microsystems 60 (2018) 15–23.
10. C. Clark, K. Fraser, S. Hand, J. G. Hansen, C. L. E. Jul, I. Pratt, A. Warfield, Live migration of virtual machines, Proceedings of the 2Nd Conference on Symposium on Networked Systems Design and Implementation 2 (3) (2005) 273–286.
11. M. Nelson, B. H. Lim, , G. Hutchins, Fast transparent migration for virtual machines, in: Proceedings of the Annual Conference on USENIX Annual Technical Conference, Anaheim, CA, 2005, pp. 472–477.
12. P. Wieder, J. M. Butler, W. Theilmann, R. Yahyapour, Service level agreements for cloud computing, in: Springer Science and Business Media, New York, NY, USA, 2011, p. 358.
13. A. Beloglazov, J. Abawajy, , R. Buyya, Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing, Future Generation Computer Systems 28 (5) (2012) 755–768.
14. E. Kumar, E. Sharma, Artificial neural networks-a study, International Journal of Emerging Engineering Research and Technology 2 (2) (May. 2014) 143–148.
15. X. Yu, M. Efe, K. O., A general backpropagation algorithm for feedforward neural networks learning, IEEE transactions on neural networks 13 (1) (7 Aug. 2002) 251–254.
16. R. S. Sutton, A. G. Barto, Reinforcement learng: An introduction, Cambridge, MA, USA: MIT Press.
17. M. L. Puterman, Markov decision processes: Discrete stochastic dynamic programming, Hoboken, New Jersey, USA: John Wiley and Sons (1998) 684.

18. A. Sözen, Future projection of the energy dependency of turkey using artificial neural network, Energy policy 37 (11) (1 Nov. 2009) 4827–4833.
19. S. Azizi, M. Zandsalimi, D. Li, An energy-efficient algorithm for virtual machine placement optimization in cloud data centers, Cluster Computing 23 (4) (2020) 3421–3434.
20. A. Khan, M. Zakarya, R. Khan, I. Rahman, M. Khan, et al., An energy, performance efficient resource consolidation scheme for heterogeneous cloud datacenters, Journal of Network and Computer Applications 150 (C) (Jan. 2020) 1084–8045.
21. J. Zeng, D. Ding, K. Kang, H. Xie, Q. Yin, Adaptive DRL-Based virtual machine consolidation in energy-efficient cloud data center, IEEE Transactions on Parallel and Distributed Systems 33 (11) (2022) 2991–3002.
22. E. Parvizi, M. Rezvani, Utilization-aware energy-efficient virtual machine placement in cloud networks using NSGA-III meta-heuristic approach, Cluster Computing 23 (4) (2020) 2945–2967.
23. Z. Li, X. Yu, L. Yu, S. Guo, V. Chang, Energy-efficient and quality-aware VM consolidation method, Future Generation Computer Systems 102 (1 Jan. 2020) 789–809.
24. M. Khan, An efficient energy-aware approach for dynamic VM consolidation on cloud platforms, Cluster Computing 24 (4) (2021) 3293–3310.
25. M. Ranjbari, J. Akbari Torkestani, A learning automata-based algorithm for energy and SLA efficient consolidation of virtual machines in cloud data centers, Journal of Parallel and Distributed Computing 113 (2018) 55–62.
26. A. Beloglazov, R. Buyya, Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers, Concurrency and Computation Practice and Experience 28 (5) (Sep. 2012) 1397–1420.
27. M. Monil, R. Rahman, VM consolidation approach based on heuristics, fuzzy logic, and migration control, Journal of Cloud Computing 5 (1) (Jul. 2016) 8.
28. Z. Han, H. Tan, G. Chen, R. Wang, Y. Chen, F. C. M. Lau, Dynamic virtual machine management via approximate markov decision process, in: IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications, San Francisco, CA, USA, 2016, pp. 1–9.
29. R. Bellman, A markovian decision process, Indiana University Mathematics Journal 6 (5) (1957) 679–684.
30. N. Rasouli, R. Razavi, H. Faragardi, EPBLA: energy-efficient consolidation of virtual machines using learning automata in cloud data centers, Cluster Computing 23 (4) (2020) 3013–3027.
31. Q. Wu, F. Ishikawa, Q. Zhu, Y. Xia, Energy and migration cost-aware dynamic virtual machine consolidation in heterogeneous cloud datacenters, IEEE Transactions on Services Computing 12 (4) (2019) 550–563.
32. H. Hallawi, J. Mehnen, H. He, Multi-capacity combinatorial ordering GA in application to cloud resources allocation and efficient virtual machines consolidation, Future Generation Computer Systems 69 (2017) 1–10.
33. M. A. H. Monil, A. D. Malony, QoS-aware virtual machine consolidation in cloud datacenter, in: 2017 IEEE International Conference on Cloud Engineering (IC2E), Vancouver, BC, Canada, 2017, pp. 81–87.
34. S. Telenyk, E. Zharikov, O. Rolik, Consolidation of virtual machines using simulated annealing algorithm, in: 2017 12th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT), Lviv, Ukraine, 2017, pp. 117–121.
35. Z. Li, C. Yan, L. Yu, X. Yu, Energy-aware and multi-resource overload probability constraint-based virtual machine dynamic consolidation method, Future Generation Computer Systems 80 (2018) 139–156.
36. S. L. Lu, J. H. Chen, Host overloading detection based on EWMA algorithm in cloud computing environment, in: 2018 IEEE 15th International Conference on e-Business Engineering (ICEBE), Los Alamitos, CA, USA, 2018, pp. 274–279.
37. M. Tarahomi, M. Izadi, M. Ghobaei-Arani, An efficient power-aware VM allocation mechanism in cloud data centers: a micro genetic-based approach, Cluster Computing 24 (2) (2021) 919–934.

38. Y. Liu, X. Sun, W. Wei, W. Jing, Enhancing energy-efficient and QoS dynamic virtual machine consolidation method in cloud environment, IEEE Access 6 (2018) 31224–31235.
39. A. Aslam, M. Kalra, Using artificial neural network for VM consolidation approach to enhance energy efficiency in green cloud, Advances in Data and Information Sciences: Springer, Singapore (2019) 139–154.
40. D. Basu, X. Wang, Y. Hong, H. Chen, S. Bressan, Learn-as-you-go with megh: Efficient live migration of virtual machines, IEEE Transactions on Parallel and Distributed Systems 30 (8) (2019) 1786–1801.
41. J. Rao, X. Bu, C. Xu, L. Wang, G. Yin, VCONF: A reinforcement learning approach to virtual machines auto-configuration, in: Proceedings of the 6th International Conference on Autonomic Computing, New York, NY, USA, 2009, pp. 137–146.
42. L. Yazdanov, C. Fetzer, VScaler: Autonomic virtual machine scaling, in: Proceedings of the 2013 IEEE Sixth International Conference on Cloud Computing, USA, 2013, pp. 212–219.
43. M. Duggan, J. Duggan, E. Howley, E. Barrett, A reinforcement learning approach for the scheduling of live migration from under utilised hosts, Memetic Computing 9 (4) (Dec. 2017) 283–293.
44. T. Ferreto, M. Netto, R. Calheiros, C. D. Rose, Server consolidation with migration control for virtualized data centers, Future Generation Computer Systems 27 (8) (Oct. 2011) 1027–1034.
45. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. D. Rose, R. Buyya, CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, Software: Practice and Experience 41 (1) (Jan. 2011) 23–50.
46. K. Park, V. S. Pai, CoMon: A mostly-scalable monitoring system for PlanetLab, ACM SIGOPS Operating Systems Review 40 (1) (2006) 65–74.
47. S. Srikantaiah, A. Kansal, F. Zhao, Energy aware consolidation for cloud computing, Cluster computing - CLUSTER 12 (1) (Nov. 2008) 10.
48. X. Fan, W. D. Weber, , L. A. Barroso, Power provisioning for a warehouse-sized computer, in: The 34th ACM International Symposium on Computer Architecture, New York, NY, USA, 2007, pp. 13–23.
49. S. Garg, A. Toosi, S. Gopalaiyengar, R. Buyya, SLA-based virtual machine management for heterogeneous workloads in a cloud datacenter, Journal of Network and Computer Applications 45 (C) (1 Oct. 2014) 108–120.
50. L. Barroso, U. Hölzle, The case for energy-proportional computing, Computer 40 (12) (Dec. 2007) 33–37.
51. K. Tsakalozos, V. Verroios, M. Roussopoulos, A. Delis, Live VM migration under time-constraints in share-nothing IaaS-clouds, IEEE Transactions on Parallel and Distributed Systems 28 (8) (25 Jan. 2017) 2285–2298.
52. W. Voorsluys, J. Broberg, S. Venugopal, R. Buyya, Cost of virtual machine live migration in clouds: A performance evaluation, in: Proceedings of the 1st International Conference on Cloud Computing, Beijing, China, 2009, pp. 254–265.
53. R. Nathuji, K. Schwan, Virtualpower: Coordinated power management in virtualized enterprise systems, ACM SIGOPS Operating Systems Review 40 (6) (2007) 265–278.
54. S. Homsi, S. Liu, G. A. Chaparro-Baquero, O. Bai, S. Ren, G. Quan, Workload consolidation for cloud data centers with guaranteed QoS using request reneging, IEEE Transactions on Parallel and Distributed Systems 28 (7) (Jul. 2017) 2103–2116.