2014 22nd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing

Energy-Efficient Virtual Machines Consolidation in Cloud Data Centers using Reinforcement Learning

Fahimeh Farahnakian, Pasi Liljeberg, and Juha Plosila
Department of Information Technology, University of Turku
Turku, Finland
{ fahfar, pakrli, juplos}@utu.fi

Abstract— **Dynamic consolidation techniques optimize resource utilization and reduce energy consumption in Cloud data centers. They should consider the variability of the workload to decide when idle or underutilized hosts switch to sleep mode in order to minimize energy consumption. In this paper, we propose a Reinforcement Learning-based Dynamic Consolidation method (RL-DC) to minimize the number of active hosts according to the current resources requirement. The RL-DC utilizes an agent to learn the optimal policy for determining the host power mode by using a popular reinforcement learning method. The agent learns from past knowledge to decide when a host should be switched to the sleep or active mode and improves itself as the workload changes. Therefore, RL-DC does not require any prior information about workload and it dynamically adapts to the environment to achieve online energy and performance management. Experimental results on the real workload traces from more than a thousand PlanetLab virtual machines show that RL-DC minimizes energy consumption and maintains required performance levels.**

Keywords— energy management; dynamic consolidation; reinforcement learning; green IT; cloud data centers

I. INTRODUCTION

As the world of computing has become very large and complex, cloud computing as a popular model delivers the computing resources on a pay-as-you-go basis. The major IT companies, such as Microsoft, Google, Amazon, and IBM are operating large-scale data centers around the world to handle the ever-increasing demand. However, the growing demand of Cloud infrastructure has considerably increased the energy consumption of data centers, which has become a critical issue. A 3% reduction in energy cost for a large company like Google can translate into over a million dollars in cost savings [1]. High energy consumption not only translates to the high cost, but also leads to high carbon emissions which are not environmentally friendly. Energy costs are increasing, data center equipment is stressing power and cooling infrastructures, and the main issue is not the current amount of data center emissions but the fact that these emissions are raising faster than any other carbon emission [2].

One of the most important reasons for energy inefficiency in data centers is the idle power wasted when servers run at a low load. Even at a very low utilization, such as 10% CPU usage, the power consumed is over 50% of the peak power [3]. Dynamic consolidation has proven to be an effective technique for power reduction in data centers by turning off idle or under-utilized servers [3- 5]. However, achieving the desired level of Quality of Service (QoS) between user and a data center is critical. Therefore, the dynamic consolidation can save energy while maintaining an acceptable QoS. The QoS requirements are formalized via Service Level Agreement (SLA) that describes such characteristics as minimal throughput, maximal response time or latency delivered by the deployed system. Moreover, virtualization is the most popular power management and resource allocation technique used by a data center. It allows a physical server (host) to be shared among multiple Virtual Machines (VMs) where each VM can run multiple application tasks. The CPU and memory resources can be dynamically provisioned for a VM according to the current resource requirements. This makes virtualization perfectly fit for the requirements of energy efficiency in a data center [4].

Reinforcement learning (RL) [7] is a machine learning paradigm that has been applied for energy management in large-scale systems. In RL, a decision-maker or agent percepts the environment and chooses an action at each state. After each action execution, the agent receives a feedback indicating the quality of the applied action. The final goal of the agent is to learn a policy for selecting the best action among all possible actions.

In this paper, we present a dynamic VM consolidation method to optimize the number of active hosts according to the current resources utilization. The method needs to make intelligent decision on when switch a host into the active or sleep power mode. For this purpose, the proposed consolidation method utilizes a learning agent. The agent learns host power mode detection policy through Q-learning which is a strong method of RL. In Q-learning [9], the agent learns on-line through experience from the environment and utilizes its knowledge to find an effective control policy for the given task. Thus, Q-learning provides a self-optimizing controller design without a prior knowledge of the environment. Results obtained from the real workload clearly show that our dynamic consolidation method based on RL outperform other dynamic consolidation schemes [10] in terms of energy consumption and SLA violation.

1066-6192/14 $31.00 © 2014 IEEE
DOI 10.1109/PDP.2014.109

The remainder of this paper is structured as follows. Related work is discussed in Section II. The Q-learning algorithm is described in Section III. The proposed system architecture and consolidation method are discussed respectively in Section IV and V. The leaning agent as a part of consolidation method explained in Section VI. The results are given in Section VII, together with a description of the simulation environment used to evaluate the performance of the proposed consolidation method. Finally, we summarize and conclude in the last section.

II. Related Work

In recent years, significant research has been done to reduce the energy cost in the cloud data centers. The pMapper [11] presents a power-aware application placement controller in virtualized heterogeneous systems for minimizing power consumption and migration cost at each time frame. In [12] a dynamic server migration is described to improve the amount of required capacity and the rate of SLA violation. It predicts variable workloads over intervals shorter than the time scale of demand variability. This work focuses on dynamic consolidation utilizing but it does not perform energy-aware placement on servers. Moreover, the sandpiper [13] implements heuristic algorithms to control VMs migration. It determines which VM to migrate from an overloaded server, where to migrate it, and a resource allocation for the virtual machine on the target server.

In some approaches, the VM consolidation have formulated as an optimization problem [13][14][15]. Although an optimization problem is associated with constraints like data center capacity and SLA. Therefore, these works utilize a heuristic method for the multi-dimensional bin packing problem as an algorithm for the workload consolidation. Data centers are bins and VMs are objects, with each data center being one dimension of the size. Algorithms solve this problem to minimize the number of bins while packing all the objects. The VirtualPower architecture [16] utilizes a power management system based on local and global policies. On the local level, the system leverages guest operating system's power management strategies. Global policy applies VMs live migration to reallocate the VMs. The PADD [17] uses an adaptive buffering scheme to determine how much reserve capacity is required. Experiments in this work show the reduced energy when the number of VMs increase.

Machine learning approaches have been investigated for resource and power management in the large-scale distributed systems such as computational grid and cloud. The task consolidation policy in [2] executes all tasks with a minimum number of resources and takes scheduling as a main role in reducing power consumption. The work used a machine-learning approach that learns from the current information of the system, such as power consumption level, CPU loads and completion time; and this contributes to improving the quality of scheduling decisions. The objective of that policy is to maximize user satisfaction without increasing power consumption. In [18] an online learning algorithm is proposed that dynamically selects different experts to make power management decisions at runtime, where each expert is a predesigned power management policy. Different experts outperform each other under different workloads and hardware characteristics.

In addition, recent studies showed the feasibility of RL approaches in resource allocation [19][20], power management [2][21]and self-optimizing memory controller [22]. In [20] allocates servers among multiple web applications dynamically using online hybrid RL to maximize the expected sum of SLA payments in each application. This hybrid approach allows the RL controller to bootstrap from existing management policies, substantially reducing learning and costliness. The effectiveness of the approach is tested in the context of a simple data center prototype. Moreover, in [21], a system level power management policy based on RL provided 24% reduction in power. It learns the optimal policy without any prior information of workload. The authors set the delay in producing an action as a performance constraint while minimizing power consumption. Considering the existing machine learning based power management techniques, the RL based learning can explore the trade-off in the power-performance design space and converging to a better power management policy.

Compared to the previous works, this work offers the following major contributions:

(1) We present a dynamic consolidation method that minimizes energy cost, while meeting the performance guarantees. In order to reduce energy consumption, this method switches an underutilized hosts to the sleep mode after all VMs migration from the host. Moreover, the method turns on the sleep hosts to avoid the SLA violation of this host when the amount of workload increases. For this purpose, we utilize a CPU usage prediction algorithm to forecast an over-loaded host. The prediction algorithm is presented in a previous work [23] to predict the short-term future resource utilization based on linear regression.

(2) As the proposed dynamic consolidation method use a learning agent, it is called Reinforcement Learning–based Dynamic Consolidation algorithm (RL-DC). The learning agent decides about the power mode of each host in a data center according the current resource usage. The agent learns the host power mode detection policy at runtime through the Q-learning technique. It is achieved by trying an action in a certain system state, and adjusting the action when this state is re-visited next time based on the penalty value that is calculated after all power mode changes. Therefore, the learning agent, as an essential part of consolidation method, can learn online the host power mode detection without prior knowledge of workloads.

(3) We apply the RL-based dynamic consolidation technique in a large-scale data center. The performance of proposed consolidation method is evaluated by CloudSim simulation on the real workload traces that is obtained from more than a thousand VMs from servers located at more than 500 places around the world. It learns the best power mode detection policy that gives the minimum energy consumption for a given performance constraint.

III. REINFORCMENT LEARNING

In Reinforcement Learning (RL), an agent can obtain the optimal solution by trail-and-error interaction with a dynamic environment without prior knowledge about the environment. In general, a framework for RL consists of [7][8]:

- State space *S*: a set of states that agent can percept from the environment.
- Action space *A*: a set of actions that agent can perform.
- A reinforcement signal *r*: a signal that agent receives form environment. In fact, this signal reflects the success or failure of the system after an action has occurred. In this paper, we consider the signal as a penalty value that the agent pays for performing an action. So, the agent aims to minimize its average long-term penalties during the learning process.

Q-learning is a one of the most popular RL methods that employed in many research areas. At each iteration of Q-learning algorithm, the agent first observes the current system state s and chooses the action *a*. After performing the action, the system moves to the next state s' and the agent also receives the reinforcement signal *r*. The signal updates the Q-value based on the following equation at the beginning of next iteration.

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha\ [r + \gamma\ min_{a\epsilon A} Q(\acute{s}, \acute{a}) - Q(s_t, a_t)] \quad (1)$$

where *Q(s,a)* represents the expected long-term cost of taking action a in state s. The learning rate α is determines in which rate the new information overwrites the old one. Learning rate can take a value between zero and one; the value of zero means that no learning takes place by the algorithm; while the value of one indicates that only the most recent information is used. The discount factor is a value between 0 and 1 which gives more weight to the penalties in the near future than the far future. The next time when an agent visits state s again, it selects the action with the minimum Q-value. The policy π for choosing the best action in state s is:

$$\pi(s) = min_{a\epsilon A} Q(s, a) \quad (2)$$

Therefore, the learning agent's goal is to find the optimal policy , mapping states to actions. The standard Q-learning algorithm has several stages as follows:

Algorithm 1. Q-learning

```
1. For each s and a, initialize Q-values to zero.
2. Observe the current state s.
3. Select action a through one of these methods and execute it:
        • Exploration or random
        • Exploitation by Equation (2)
4. Receive reinforcement signal r.
5. Observe the new state s' and update Q(s, a) using Equation (1).
6. $s \leftarrow s'$.
7. Go back to step1.
```

There are two ways for selecting an action from the possible actions in every state:

- Exploration or random action selection: at the beginning of learning, optimal actions are not chosen yet. Therefore, the agent chooses an action randomly.
- Exploitation: actions are selected according to learned policy π.

IV. SYSTEM MODEL

We consider a large-scale data center as a resource provider that consists of m heterogeneous physical nodes. Each node has a processor, which can be multi-core, with performance defined in Millions Instructions Per Second (MIPS). Besides that, a host is characterized by the amount of memory, processing capacity and network bandwidth. Several users submit requests for provisioning of *n* VMs characterized by requirements to CPU performance, RAM, network bandwidth and disk storage. Initially, the VMs are allocated according to the requested characteristics assuming 100% CPU utilization. The length of each request specifies by millions of instructions (MI). As the VMs experience dynamic workloads, the CPU usage by a VM arbitrarily varies over time. In order to reduce SLA violation and energy consumption, VMs consolidate on the minimum number of hosts according to the current requested resources. When the utilization of resources on a host is low, all VMs reallocate to other hosts and the under-loaded host switch to the sleep mode. In addition, some VMs on a host must be migrated in order to reduce SLA violation while the host becomes overloaded. The quality of VM consolidation algorithm can improve using a learning agent to determine the host power mode (sleep or active). The agent learns the efficiency of resource allocation and energy consumption based on Q-learning.

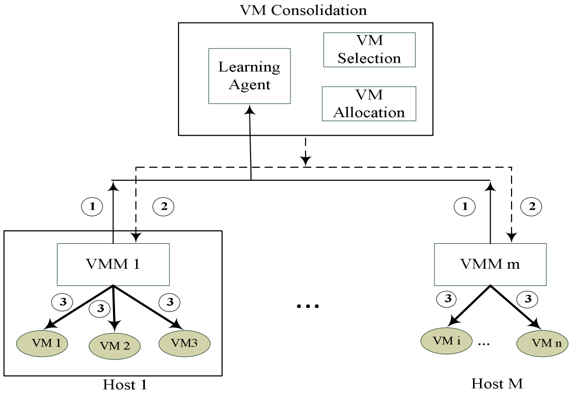

Figure 1. The system model

In general, the sequence of proposed dynamic consolidation operations is as follows (Figure 1):

(1) Observing the current host status by a learning agent.

The learning agent first collects the information about the current state of hosts from VM Managers (VMMs). This information represents the current total utilization of a host. Then, the agent decides about the power mode of each host based on Q-learning.

(2) Sending the allocation map to VMM.

The proposed VM consolidation method optimizes the VMs placement depending on the agent's about the power mode. If the specified host power mode is sleep, all VMs from the host must migrate to other hosts. Therefore, the VM allocation selects a host to allocate the VM. Moreover, some VMs migrate from a host to other hosts if the host mode is active and it becomes over-loaded. So, the VM consolidation uses the VM selection policy to choose which VM to migrate from the over-loaded host. Finally, the consolidation algorithm generates an allocation map and sends it to VMM. The allocation map determines which VM should be allocated to which host.

(3) VMs migration commands.

The VMMs perform reallocation of some VMs according to the received allocation map from VM consolidation. So, the VMM sends the migration commands to VMs that should be migrated to other hosts.

V. Reinforcment Learning-based Dynamic Consolidation Method

In this paper, a dynamic consolidation method is proposed in order to reduce energy cost and SLA violation of data center and named Reinforcement Learning–based Dynamic Consolidation (RL-DC) algorithm. RL-DC can dynamically adapt a number of active hosts to the variable workload. An important part of the consolidation algorithm, is to decide whether (1) additional host are required to provide efficient resource utilization with an increasing workload, or (2) redundant hosts can be put sleep to save energy or (3) the current amount of hosts is sufficient. To make this decision, a learning agent is assumed as an essential part of RL-DC. The details of the proposed consolidation algorithm are presented in Algorithm 2.

Algorithm 2. RL-DC

```
1.  for each host h do
2.      powerMode ← learningAgent(h)
3.      if (powerMode = sleep && currentMode ≠ sleep)
4.          for each VM on host h do
5.              selectedHost ← VMAllocation(VM)
6.              migrate VM to selectedHost
7.          end for
8.          switch the host h to the sleep mode
9.      end if
10.     if (powerMode = active)
11.         if (currentMode ≠ active)
12.             switch the host h to active mode
13.         end if
14.         predictUtil ← LiRCUP(h,CurrUtil)
15.         while( predictedUtil > AvailableUtil) do
16.             selectedVM ← VMSellection(h)
17.             selectedHost ← VMAllocation(selectedVM)
18.             migrate selectedVM to selectedHost
19.         end while
20.     end if
21. end for
```

The agent first percepts the information about the current power consumption, total CPU utilization and power mode of hosts at beginning a time slot. The time between two iteration of the consolidation algorithm is called the time slot. Then, the host power mode (active or sleep) in the next time slot based on this information and its experience of previous host state is determined by the agent (line 2). The RL-DC algorithm optimizes the resource allocation according to the specified power mode of hosts. While the learning agent decides a host should be switched to the sleep, RL-DC migrate all VMs from the host to other hosts (line 3-9). The VM allocation algorithm (Algorithm 3) selects a host to allocate VM from the host that must be switched to the sleep mode (line 5). Therefore, the energy cost and CO2 emissions can be reduced in a data center by switching the under-loaded hosts to the sleep mode. Moreover, when the host power mode is decided to be active and the current mode is not the active mode; the host will be switched to the active mode (line 11-13). RL-DC employs a prediction method, LiRCUP, to avoid the SLA violation (line 14). Based on the past CPU utilization values in a host, LiRCUP approximates a function based on the linear regression [23]. The function can forecast the short-term utilization of host by considering on the historical data of usage. If the predicted usage exceeds of available host usage, the host becomes over-loaded. So some VMs on the host must migrate to other hosts before a SLA violation happen (while loop). The VM selection algorithm selects which VM should be migrating to other hosts (line 16). The selected VM reallocate to the host that is chosen with the VM allocation algorithm (line 17 and 18).

When RL-DC needs to select a host for allocating a VM, it uses the VM allocation policy. This policy first finds the hosts are not be over-loaded at current and next times after VM allocation (NotOverLoadedList). This means these hosts have free resources that can be shared among VM. Then, it chooses a host from *NotOverLoadedList* so that the power increasing is minimize after VM allocation (Selected host). Algorithm 3 describes the proposed VM allocation algorithm.

Algorithm 3. VM Allocation

```
Input: VM
Output: selectedHost
1.  for each host from hostList do
2.      predictUtil ← LiRCUP(h,CurrUtil)
3.      if( availableUtil > predictedUTil+ RequestedUtilByVM)
4.          NotOverLoadedList ← h
5.      end if
6.  end for
7.  minPower ←MAX
8.  for each host from NotOverloadedList do
9.      power ←estimatePower(host, VM)
10.     if (power <minPower)
11.         selectedHost←host
12.         minPower← power
13.     end if
14. end for
15.return selectedHost
```

Since we compared the proposed consolidation with four algorithms in [6], we assumed the same VMs selection policy on these algorithms. This policy is named Minimum Migration Time (MMT) because it selects a VM for migration that requires the minimum migration time than other VMs on the host. The migration time is calculated with dividing the memory assigned to the VM, by the available network bandwidth between the original and the target host. Since all network links has 1GBPS bandwidth in our simulation, only the amount of RAM utilized on the VM is considered as migration time measure.

Table I. The power consumption at different load levels in Watts

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

VI. LEARNING AGENT

An efficient consolidation method should reduce the number of active hosts according to the current workload. It needs to make intelligent decisions on when to put the hosts into the sleep or active power mode. For this reason, we propose a learning agent as an important part of RL-DC. The agent specifies the host power mode based on past history of data and improves itself as the environment changes. For this purpose, it learns online the host power mode detection policy upon incoming requests and adjusts the policy accordingly through Q-learning. The agent first perceives the current state space consisting status of all hosts at the beginning of the current time slot *t*. The state space set *S* includes of *m* members, where *m* is the number of host in the data center. Each element of *S* represents the current total requested CPU utilization of all VMs on a host. Since recent studies show the CPU utilization has a linear relationship on power consumption, when dynamic voltage and frequency scaling is applied. The power consumption by servers can be accurately described by a linear relationship between the power consumption and CPU utilization [10] [25]. Therefore, the resource capacities of the host and resource usage by VMs are characterized by a single parameter, the CPU performance.

The agent performs an action based on the observed state. The action space is defined as a set $A= \{PM_{h1}, PM_{h2},…,PM_{hn}\}$, with *n* members that indicate the all power mode of hosts. Each element of set *A*, PM_{hi}, represents the power mode of host *i* (active, sleep) on next time slot *t+1*. The RL-DC switches each host to the specified power mode based on the agent decision. Then, the agent calculates the action penalty as the reinforcement signal after changing all power modes at the beginning of the time slot *t+1*. As the main objective of RL-DC is to minimize the energy cost and SLA violation values, calculating the penalty value *P* consisting of two values: the SLA violation penalty and the energy consumption penalty.

$$P_t = P_t(SLA) + P_t(power) \qquad (3)$$

A. The SLA violation penalty

Achieving desirable QoS requirements is extremely important for a Cloud computing environment. The QoS requirements are commonly defined in term of SLA that describes such characteristics as throughput or response time. Since these characteristics can change for different applications, it is necessary to define a metric that is independent of workload and it can be used to evaluate the SLA delivered to any VM deployed in a data center. We computed the SLA violation as the difference between the requested MIPS (U_r) by all VMs and the actually allocated MIPS (U_a) over a time slot.

$$SLA_t = \sum_{i=1}^{n} (U_{ri} - U_{ai})$$

where *n* is the number of VMs. The penalty of SLA violation is calculated by dividing the SLA value after action a_t execution by the value of SLA before performing an action.

$$P_t(SLA) = \frac{SLA_{t+1}}{SLA_t} \qquad (4)$$

If the current SLA violation is less than the previous time slot ($SLA_{t+1} < SLA_t$), the value of penalty is small than one. It means the chosen action by the agent is a proper action to minimize the SLA violation.

B. The power consumption penalty

Our simulation environment is an extension of the CloudSim 3.0 toolkit [26]. We have selected two server configurations in CloudSim: HP ProLiant ML110 G4 (Intel Xeon 3040, 2 cores -1860 MHz, 4 GB), and HP ProLiant ML110 G5 (Intel Xeon 3075, 2 cores- 2660 MHz, 4 GB). Table I illustrates the power consumption characteristics of the selected servers in the simulator. The reason why we have not chosen servers with more cores is that it is important to simulate a large number of servers to evaluate the effect of consolidation. Nevertheless, dual-core CPUs are sufficient to evaluate resource management algorithms designed for multi-core CPU architectures [6].

The power consumption penalty is measured by dividing the power consumption value at current time slot by the power consumption of previous time slot. So, P_t *(power)* represents the total power consumption penalty of *m* hosts.

$$P_t(power) = \sum_{i=1}^{m} \left(\frac{power_{\ t+1}}{power_t}\right) \qquad (5)$$

All of VM consolidation and allocation are completed at the beginning of next time slot *t+1*. Then the Q-value that is related for each pair of action and state of time slot is updated through the total penalties value (P_t). We suppose a 50-50 weight is assigned to old and new information (= 0.5). Thus simple weighting assignment has performed the best compared to other cases, so that a value of 0.7 is assigned to . Therefore, the equation (1) is rewritten as:

$$Q(s_t, a_t) = Q(s_t, a_t) + 0.5\ [P_t + 0.7\ min_{a\epsilon A} Q(\acute{s}, \acute{a}) - Q(s_t, a_t)]$$

The Q-value of state-action pair, Q(s,a) represents the expected total power and SLA violation caused by the action *a* taken in the state *s*. When the agent observes the state *s* next time, it selects the power mode of hosts that is provided the minimum Q-value. The best action that has the lowest Q-value (SLA violation and power consumption) will be selected by the learning agent. So, the proposed dynamic consolidation algorithm can achieve the performance and power trade-off in cloud data centers. The pseudo code of learning agent algorithm can be summarized in six steps:

Algorithm 4. Learning Agent

```
1.Percepts the current state $s_t$ at the beginning time slot t.
2.Select an action $a_t$ = (active/sleep) based on static threshold or by using Equation (2) (agent knowledge).
3. Calculate the SLA violation penalty $P_t$(SLA)using Equation( 4).
4. Calculate the power consumption penalty $P_t$(power) using Equation( 5).
5. Compute the total penalty $P_t$ after all power mode changes using Equation( 3).
6. Update the $Q(s_t,a_t)$ value using Equation( 6).
```

During the beginning of the learning process and whenever the agent has not visited the current state before, an action is based on static lower threshold. The threshold is more efficient than the random selection in the standard Q-learning. If the host utilization exceeds of 40% of the total amount of CPU available capacity on the host, the agent set the sleep mode of host to active. Otherwise, the host is under-loaded and it should be switch to the sleep mode.

VII. Simulation Results

To evaluate the efficiency of our approach, implementations have been performed on the CloudSim toolkit. CloudSim is becoming increasingly popular in the cloud computing community due to it support for flexible, scalable, efficient, and repeatable evaluation of provisioning policies for different applications [26]. We simulated a data center comprising 800 heterogeneous hosts. The number of VMs depends on the type of workload: random or real workload. In random workload, the users submit requests for provisioning of 800 heterogeneous VMs that fill the full capacity of the simulated data center. Each VM runs an application with a variable workload, which is modeled to generate the utilization of CPU according to a uniformly distributed random variable. The application runs for 150,000 MI that is equal to 10 minutes of the execution on 250 MIPS CPU with 100% utilization. In real workload, the number of VMs on each day is specified in Table II. Real workload data is provided as a part of the CoMon project, a monitoring infrastructure for PlanetLab [27]. In this project, the CPU utilization data is obtained from more than a thousand VMs from servers located at more than 500 places around the world. Data is collected every five minutes and is stored in a variety of files. We selected five days from the workload traces collected during April 2011 of the project. During the simulation, each VM is randomly assigned a workload trace from one of the VMs from the corresponding day. The characteristics of the VM types correspond to Amazon EC2 instance type with the only exception that all the VMs are single-core, which is explained by the fact that the workload data used for the simulations come from single-core VMs.

Table II. The number of VMs in the real workload

Date	Number of VMs
3 April	1463
9 April	1358
11 April	1233
12 April	1054
20 April	1033

We compared the RL-DC method with three algorithms in [10] which are presented heuristics for dynamic reallocation of VMs with workloads originating from web applications and online services. The main idea of these algorithms is to set upper and lower utilization thresholds and keep the total CPU utilization of a node between these bounds. When the upper bound is exceeded, VMs are reallocated for load balancing and when the utilization of a host drops below the lower bound, VMs are reallocated for consolidation. The algorithms adapt the utilization threshold dynamically based on the Median Absolute Deviation (MAD), the Interquartile Range (IQR) and Local Regression (LR) approach to estimate the CPU utilization. In addition, we consider the static threshold method (THR) in [10] that monitors the CPU utilization and migrates a VM when the current utilization exceeds of 80% of the total amount of CPU available capacity on the host. We supposed two metrics for performance evaluation of proposed dynamic VM consolidation process based on the Q-learning.

A. Average SLA violation percentage

This metric represents the percentage of average CPU performance that has not been allocated to an application when requested, resulting in performance degradation [5]. It is calculated by Equation (7) as a fraction of the difference between the requested by all VMs and the actually allocated MIPS relatively to the total requested MIPS over the life-time of the VMs , where n is the number of VMs.

$$SLA = \frac{\sum_{i=1}^{n} \int U_r(t) - U_a(t)dt}{\sum_{i=1}^{n} \int U_r(t)dt} \qquad (7)$$

Table III illustrates the SLA violation levels caused by the RL-DC, THR, MAD, IQR and LR methods in the random workload. RL-DC can reduce the percentage of SLA violation rate more efficiently than other techniques. The obtained results can be explained by the fact that the RL-DC avoids the SLA violation by the over-loaded host prediction. Moreover, it learns to minimize the SLA violation by considering the current resource requirements.

Table III. Average SLA violation percentage in the random workload

RL-DC (%)	THR(%)	MAD(%)	IQR(%)	LR(%)
8.65	12.75	10.75	10.35	14.89

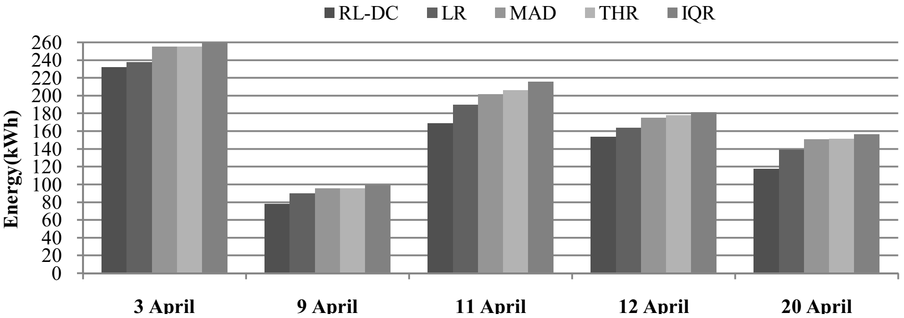

Figure 2. Energy consumption by RL-DC and benchmark methods in the real workload

Table IV demonstrates the percentages of average SLA violation for the real workload. The results show the RL-DC lead to significantly less SLA violation than other four benchmark algorithm. The reason is that the RL-DC learn to switch the host to the active mode before a SLA violation happens. Moreover, VM allocation algorithm allocates the VM on a host that it does not becomes over-utilized in short-time.

Table IV. Average SLA violation percentage in the real workload

Date	RL-DC(%)	THR(%)	MAD(%)	IQR(%)	LR(%)
3 April	8.43	10.07	10.11	10.05	10.05
9 April	8.62	10.25	10.05	10.01	10.16
11 April	9.05	10.08	10.10	10.01	10.41
12 April	9.65	10.27	10.04	10.17	10.45
20 April	9.80	10.75	10.49	10.35	11.28

B. *Energy consumption*

This metric is the total energy consumption by the physical resources of a data center caused by the application workloads. Table I illustrates the power consumption characteristics of the selected servers in the simulator. Since the utilization of the CPU may change over time due to the workload variability. Thus, the CPU utilization is a function of time and is represented as *U(t)*. Therefore, the total energy consumption by a physical node (E) can be defined as an integral of the power consumption function over a period of time as shown in Equation (8).

$$E = \int_{t_0}^{t_1} P\big(U(t)\big)dt \qquad (8)$$

Figure 2 shows the proposed dynamic VM consolidation based on RL can bring higher energy saving in comparison to other policies without learning of previous information. By enabling the learning algorithms presented in the RL-DC, a significant reduction of the energy consumption of 12.5%, 19.4%, 22.6% and 28.5% can be reached by comparing LR, MAD, THR and IQR in the real workload on 11 April, respectively.

In addition, Figure 3 shows the RL-DC consumes less power than other benchmarks algorithms in the random workload. The proposed method can learn to detect an under-utilized host through the learning agent and allocated all VMs to other hosts for switching it to the sleep mode. Moreover, VMs allocate to hosts which are increases the least power consumption in a data center. So, with learning ability that is presented in this paper, we can achieve a significant reduction of 15.6%, 33.6%, 44.3% and 46.8% of the energy consumption can be reached compered to LR, THR, MAD and IQR methods in the random workload, respectively.

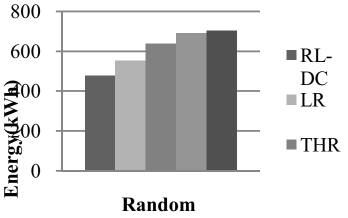

Figure 3. Energy consumption by RL-DC and benchmark methods in the random workload

VIII. Conclusion

In this paper, we presented a dynamic consolidation method to reduce power consumption and SLA violation in the cloud data centers. It employs the reinforcement learning approach to learn the host power mode detection policy without prior knowledge of the environment and workload. Therefore, the method can adapt the number of active hosts to the current resources requirements. Compared with the existing dynamic consolidation methods in CloudSim simulation, the proposed reinforcement learning-based dynamic consolidation method is able to minimize energy cost and SLA violation rate efficiently.

References

[1] G. A. Qureshi, R. Weber, H. Balakrishnan, J. Guttag, and B. Maggs, " Cutting the electric bill for Internet-scale systems", Proceedings of the ACM SIGCOMM 2009 conference on Data communication, pp.123-134, 2009.

[2] J. Ll. Berral, I. Goiri1, R. Nou, F. Julia , J. Guitart, R. Gavaldà and J. Torres; "Towards energy-aware scheduling in data centers using machine learning", Procedings of the 1st Internatinal Conference on Energy-Eficient Computing and Networking, pp. 215-224, 2010.

[3] G. CHEN, et al. "Energy-aware server provisioning and load dispatching for connection-intensive internet services", Procedings of

the 5th USENIX Symposium on Networked Systems Design and Implementation, pp. 337-350, 2008.

[4] J. Shuja, S. Ahmad Madani, K. Bilal, Kh. Hayat, S. Ullah Khan, Sh. Sarwar, " Energy-efficient data centers", Computing 94(12): pp. 973-994, 2012.

[5] A. Beloglazov, J.Abawajy, R.Buyya," Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing", Journal of Future Generation Computer Systems, vol.28, pp.755-768, 2012.

[6] L. Deboosere, B. Vankeirsbilck, P. Simoens, F. D. Turck, B. Dhoedt, and P. Demeester. "Efficient resource management for virtual desktop cloud computing", The Journal of Supercomputing , pp.741-767, 2012.

[7] R.S.Sutton, A.G.Barto, "Reinforcement Learning: AnIntroduction", MIT Press,1998.

[8] L.P.Kaelbling, "Reinforcement Learning: A Survey, Journal of artificial Intelligence Research", pp.237-285, 1996.

[9] C.J.Watkins, "Q-Learning, Machine Learning" , pp. 279-292, 1992.

[10] A. Beloglazov and R. Buyya, "Optimal Online Deterministic Algorithms and Adaptive Heuristics for Energy and Performance Efficient Dynamic Consolidation of Virtual Machines in Cloud Data centers", Concurrency and Computation: Practice and Experience (CCPE), Vol.24, pp.1397-1420, 2012.

[11] A. Verma, P. Ahuja, and A. Neogi, "pMapper: power and migration cost aware application placement in virtualized systems", Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware, p.p 243–264, 2008.

[12] N. Bobroff, A. Kochut, and K. Beaty, "Dynamic placement of virtual machines for managing SLA violations", Proceedings of the 10th IFIP/IEEE Intl. Symp. on Integrated Network Management (IM), pp.119–128, 2007.

[13] Wood, P. J. Shenoy, A. Venkataramani, M. S. Yousif, "Sandpiper: Black-box and gray-box resource management for virtual machines", Journal of Computer Networks, vol. 53, pp. 2923–2938, 2009.

[14] Y. Ajiro and A. Tanaka, "Improving packing algorithms for server consolidation," Proceedings of the International Conference for the Computer Measurement Group (CMG), pp. 399–407, 2007.

[15] M. Wang, X. Meng, L. Zhang, "Consolidating Virtual Machines with Dynamic Bandwidth Demand in Data centers", Proceedings of IEEE INFOCOM 2011 MINI-CONFERENCE, pp. 71-75, 2011.

[16] R. Nathuji and K. Schwan, "Virtual Power: Coordinated Power Management in Virtualized Enterprise Systems", Proceedings of the 22st ACM Symposium on Operating Systems Principles (SOSP'07),pp. 265–278 , 2007.

[17] M. Y. Lim, F. Rawson, T. K. Bletsch, V. W. Freeh. "PADD: Power-Aware Domain Distribution", Proceedings of the 29th International Conference on Distributed Computing Systems (ICDCS), pp. 239-147, 2009.

[18] G. Dhiman and T. S. Rosing, " System-level power management using online learning", Proceedings of the Computer-Aided Design of Integrated Circuits and Systems (CADICS), pp. 676–689, 2009.

[19] J. Rao, X. Bu, C.-Z. Xu, L. Wang, and G. Yin. " Vconf:a reinforcement learning approach to virtual machine autoconfiguration", Proceedings of the 6th International Conference on Autonomic Computing (ICAC), pp. 137-146, 2009.

[20] G. Tesauro, N. K. Jong, R. Das, and M. N. Bennani, " A hybrid reinforcement learning approach to autonomic resource allocation", Proceedings of the the IEEE International Conference on Autonomic Computing (ICAC), pp. 65–73, 2006.

[21] Y. Tan, W. Liu, and Q. Qiu, "Adaptive power management using reinforcement learning", Proceedings of the International Conference on Computer-Aided Design (ICCAD '09), pp 461–467, 2009.

[22] E. Ipek, O. Mutlu, J. F. Martinez, and R. Caruana. "Self-optimizing memory controllers: A reinforcement learning approach", Proceedings of the 35th Annual International Symposium on Computer Architecture (ISCA), pp.39-50, 2008.

[23] F. Farahnakian, P. Liljeberg, and J. Plosila, "LiRCUP: Linear Regression based CPU Usage Prediction Algorithm for Live Migration of Virtual Machines in Data Centers", Proceedings of the 39th Euromicro Conference on Software Engineering and Advanced Applications(SEAA), pp.358-364, 2013.

[24] X. Fan, WD. Weber, LA. Barroso, "Power provisioning for a warehouse-sized computer", Proceedings of the 34th Annual International Symposium on Computer Architecture (ISCA 2007), pp.13– 23, 2007.

[25] D. Kusic, JO. Kephart, JE. Hanson, N. Kandasamy, G. Jiang, " Power and performance management of virtualized computing environments via lookahead control", In Proceedings of the International Conference on Autonomic Computing (ICAC), pp.3-12, 2008.

[26] RN. Calheiros, R. Ranjan, A. Beloglazov, CA, F. De Rose, R. Buyya, " Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms", Journal of Software: Practice and Experience (SPE), pp.23–50, 2011.

[27] M. Stone, " Cross-validatory choice and assessment of statistical predictions", Journal of the Royal Statistical Society, pp. 111–147, 1974.