Modeling of the Data Center Resource Management Using Reinforcement Learning

Sergii Telenyk
Cracow University of Technology, Poland
stelenyk@pk.edu.pl

Eduard Zharikov, Oleksandr Rolik
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine
zharikov.eduard@acts.kpi.ua, o.rolik@kpi.ua

***Abstract*—Cloud data centers are most dynamic systems in a modern digital world. To deliver the high-performance and fault-tolerant IT services to end users effectively it is necessary to develop new methods for data center resource management while adapting to the emergence of new requirements. In this paper, the authors refine and evaluate the previously proposed method for cloud data center resource management based on the reinforcement learning approach. The proposed method takes into account the power consumption and the number of SLA violations in the management policy. The power consumption is managed by switching physical servers to active or sleep state depending on current utilization of three resources: CPU, memory, and network bandwidth. The proposed reinforcement learning agent allows to determine the optimal policy for managing the physical servers without creating an environment model and preliminary information about the workload. The evaluation results show that the proposed method allows to decrease the SLA violation time, to serve more VM schedule requests when the number of VMs is changing frequently, and to decrease the utilization of data center network due to decreased number of migrations.**

Keywords—data center; resource management; learning agent; energy efficiency

I. INTRODUCTION

During the last five years, the number of cloud services and clients have grown significantly, and the complexity of the underlying software and hardware infrastructure is also increasing. Modern challenges lead to development of new methods for data center resource management by adapting to the emergence of new information services. A wide range of the modern information services, applications, and computing resources are almost always provided by cloud data centers. The cloud data center is a complex system to deliver high-performance and fault-tolerant IT services for users and tenants using a utility computing concept [1].

Cloud computing is presented by distributed type of computing resource consisting of data centers, infrastructure layer facilities, and physical machines (PMs) with each hosting several virtual machines (VMs). The VMs can be provisioned and released dynamically and are also presented to users as processing and storage resources based on service level agreements (SLAs). The SLA is used as a formal contract between a cloud service provider and a user to ensure service quality [2]. Cloud service providers ensure the SLA to maximize the revenue and improve the user satisfaction. Achieving the desired QoS requirements is extremely important for the cloud service providers. The QoS requirements are usually defined in terms of SLA that describe characteristics such as response time and scheduling delays.

There is a significant effort of research in the data center resource management field including resource provisioning, allocation, scheduling, and capacity planning [3]. The workload in the cloud data center can change over time. This requires to periodically solve an optimization problem to provision new and to reallocate existing VMs. The virtual machine consolidation problem [4] is the subject to many constraints such as the VM resource requirements, security requirements, availability requirements, and others. Thus, there is a need to develop models and algorithms for the VM consolidation and migration management when processing data center workloads. Such solutions need to be based on the assessment of the state of the data center resources and workloads.

In [5], the authors propose the reinforcement learning method to solve the dynamic VM placement problem. Reinforcement Learning (RL) [6] is one of the machine learning methods in which the agents perform actions in order to minimize the total penalty as a result of each action. To calculate the penalty, an SLA violation indicator and a power consumption indicator are taken into account. The purpose of the agent is to get as little penalty as possible choosing any management action according to the management policy. In this paper, the authors refine the proposed method by elaborating of the core algorithm, by decreasing the size of state space, by decreasing the number of the resource utilization intervals and the number of the state space element attributes. Besides, the authors evaluate refined method by running simulations using CloudSim [7] and Bitbrains traces [8].

The remainder of the paper is organized as follows: in Section II the related work review is presented, Section III describes the refined reinforcement learning method for data center resource management, Section IV describes the model and algorithms of the method of dynamic VM consolidation, in Section V the results of simulation are presented, and Section VII concludes the paper.

II. RELATED WORK

Many previous studies concerning the data center resource management with different objectives [9], [10], [11] focus on three main objectives such as (i) ensuring the SLA between a cloud service provider and a user (ii) reducing the power consumption of the data centers, and (iii) reducing the operational costs of managing data center services.

978-1-5386-6611-1/18/$31.00 ©2018 IEEE

Significant attention for research in recent years have highlighted the data center resource management problem. Several studies proposed various solutions to find optimal resource allocation with the objective to minimize the number of PMs used while complying the SLA [12].

In [13], the authors perform a detailed analysis of the problem of energy efficient and performance efficient dynamic consolidation of VMs. The authors analyze online, offline, deterministic and dynamic VM consolidation problem and propose adaptive heuristics for dynamic VM consolidation process. The disadvantage of the proposed methods is that the number of simultaneous migrations per PM is not limited. Besides, the power consumption of PMs in the sleep mode and the power consumption during switching from the sleep mode to the active mode are not considered.

It should also be noted that several studies such as [14], [15], [16], [17] propose methods and algorithms that are evaluated using simplified workload namely the CPU utilization reported in PlanetLab [18]. At the same time, such resources as memory and network utilization are not considered in proposed models. Moreover, many simulation scenarios use outdated PM and VM configurations.

In some previous studies [19], [20], [21], [22], [23] it is reported that the time of switching a PM from the sleep mode to active mode can reach 200 seconds. Moreover, during that time the power consumption of a PM reaches the peak rate. Thus, the transition modes in cloud data center have a significant impact on the power consumption of a modern data center and would not be ignored.

One of the promising approach for optimal allocation of cloud resources is Reinforcement Learning [6]. In [24], the management of virtual resources in a cloud environment is considered as a problem of automatic control using the RL approach. The authors used the Q-learning method [25] to manage the number of VMs that provide cloud service. The proposed algorithm retains the value of the "action-reward" pairs as historical values and uses them to decide whether to change the number of virtual machines during cloud resource management. But the proposed approach does not consider the placement of virtual machines on physical machines.

In [26], the authors propose the online hybrid RL algorithm to dynamically allocate servers among multiple web applications. The authors combine RL with queuing models in a hybrid approach, in which RL algorithm is trained offline on data collected while a queuing model policy controls the data center resources. But the proposed algorithm is limited by orientation on the web applications and is not applicable to virtual environments.

To get the balance between QoS revenue and power consumption, in [17], the authors propose the reinforcement learning-based adaptive resource management algorithm. The proposed algorithm does not need to assume prior distribution of resource requirements and is robust in actual workload. The goal of the proposed approach is to find the best balance between QoS revenue and power consumption. But the proposed approach considers only CPU utilization and does not consider other PM resources. Furthermore, the proposed algorithm does not account influence of the VM's memory changes on the migration overhead.

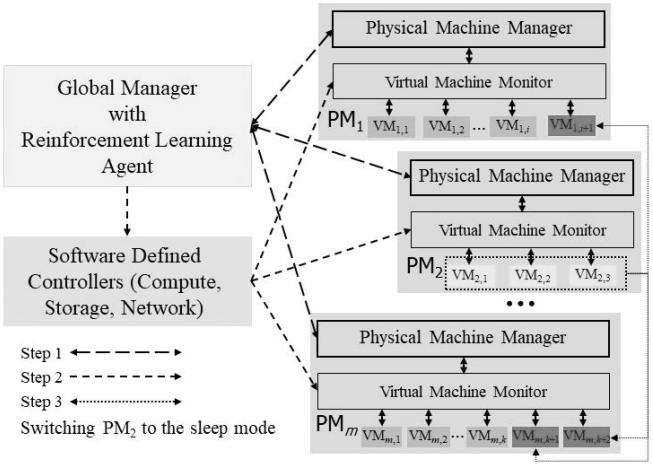

Fig. 1. The data center model

In [27], the authors solve dynamic server resource provisioning problem using post decision state learning-based algorithm with fast convergence. The authors investigate the performance effect of the state-space size on the performance of the proposed method. As a result, the authors conclude that a larger state-space size decreases the convergence rate of the proposed method. Some new technics are also introduced to derive the learning algorithm having an accelerated convergence compared with the conventional Q-learning algorithm. However, the proposed method accounts only homogeneous servers. Furthermore, the authors use the model of the workload arrival distribution that cannot be accurately estimated in production environment with mixed workload.

III. The Proposed Approach

The cloud data center consists of a set of *m* PMs, each of which is characterized by hardware and operating platforms and has a fixed number of resources. The data center model is shown in Fig. 1. Each PM is characterized by the CPU, the amount of RAM, and the network bandwidth. Each CPU can be multi-core, with the productivity measured in millions of instructions per second (MIPS). The user resource requirements and workloads can change over time by requesting services or jobs that spans one or multiple VMs. This requires provisioning of a new and reallocating of existing VMs in the data center.

The Global Manager (GM) is the main module of the cloud data center that implements the proposed reinforcement learning method. In a broad sense the GM manages data center virtual and physical resources and allows to select a variety of management policies in order to adapt to the impact of external factors such as workload change, update installation, the use of a new software and hardware platforms, changes in the data center structure and changes in the performance of the services provided. The GM also performs the PM and VM state management, VM scheduling, and VM consolidation. One of the goals of the GM is to place the VMs at the minimal number of PMs to reduce data center power consumption and to decrease SLA violations.

In [5], to manage data center resources, the authors propose a model-free version of the learning agent, based on the Q-learning method [25]. In this paper, the Q-learning algorithm presented in [5] is elaborated upon and refined according to the results of experiments. The learning agent generates close to optimal management actions by interacting with the data center controllers without any prior information about the workload incoming to VMs.

At each step of the management process, the agent observes the current system state $s_t \in S$ and chooses an action $a_t \in A$ that impacts the data center subsystems. After performing the action, the system moves to the next state $s_{t+1} \in S$ and the agent obtains a penalty p_t. At the beginning of the next management step t+1 the learning agent observes the current system state $s_{t+1} \in S$ and updates the Q-value to the t-th step of management process. The update of the Q-value is defined as follows:

$$Q(s_t, a_t) = (1-\alpha)Q(s_t, a_t) + \alpha[p_t + \gamma \min_{a \in A} Q(s_{t+1}, a_{t+1})], \quad (1)$$

where $Q(s_t, a_t)$ is an expected long-term penalty for performing the action $a_t \in A$ at t-th step; α is a learning rate, the coefficient indicating how fast the data about penalties in new states will be taken into account in the next step, $\alpha \in [0,1]$; γ is a discount factor which determines the importance of the future penalties, $\gamma \in [0,1]$; $\min_{a \in A} Q(s_{t+1}, a_{t+1})$ is an estimation of the Q-value for performing the action $a_{t+1} \in A$ at t+1 step.

If $\alpha = 0$ then the agent does not learn to improve future actions. If $\alpha = 1$ then the agent uses the data on the results of the latest management actions. If $\gamma = 0$ then the agent takes into account only the latest penalties. In case of $\gamma = 1$ the agent aspires to a long-term minimal penalty. When the agent receives the state $s_t \in S$ again it will select the action with a minimum Q-value. A management policy π of choosing the best action in the state $s_t \in S$ is defined as follows:

$$\pi(s_t) = \arg\min_a (Q(s_t, a)). \quad (2)$$

Thus, the goal of the learning agent is to find the best mapping policy $S \to A$ that minimizes expected long-term penalty for performing management actions. To choose management action, the learning agent may use one of two techniques: (i) the random action choice can occur at the beginning of the management process along training period, (ii) the choice of action is defined by the policy π.

During the resource management process, the learning agent converges to the best control policy by revising the existing management policy in response to data center state changes. The main disadvantage here is the speed of convergence may not satisfy the SLA requirements or may not minimize power consumption of data center. The reason is the learning agent needs to perform some number of exploration actions adopting to new environment states. Another disadvantage is the big size of the state space. In this paper, the authors decrease the state space size compared to [5] by decreasing the number of the resource utilization intervals and by decreasing the number of the state space element attributes.

The learning agent acts as follows.

Step 1. The observation of the PM's current state. The learning agent receives information about the current state of resource usage of each PM from the Physical Machine Managers (PMM) and selects the operation mode for each PM based on the Q-learning algorithm. The PMM calculates the available resource capacity of the PM such as central processor (*CPU*), memory (*RAM*), network bandwidth (*BW*) and their predicted utilization.

Step 2. The sending control impacts to virtual machine monitors (VMM). The VM reallocation occurs depending on the agent's decision about the future state of each PM. If the learning agent chooses the sleep mode for *i*-th PM, then all VMs on *i*-th PM are to migrate to other PMs. The choice of a VM for migration from overloaded PM occurs in accordance with the management policy. The authors propose to use the policy that chooses a VM that has minimal RAM size.

Step 3. Initiating the VM migration commands. The VMM sends migration commands to the selected VMs which must migrate to some other PMs.

IV. The Use Of Reinforcement Learning For VM Consolidation

In this Section, the method of dynamic VM consolidation proposed by the authors in [5] is refined and briefly described.

The aim of the proposed method is to reduce the number of SLA violations and data center power consumption. The SLA includes requirements and limitations on the quality of service provided: response time, VM scheduling delay, availability, etc. The VM scheduling delay is important in scale-out scenarios when the workload increases and additional VM instances are needed. The SLA is fulfilled when each customer obtains all the performance required by applications inside the VM or VM group.

The flowchart of VM consolidation algorithm based on reinforcement learning is illustrated in [5]. The result of the algorithm is the list of virtual machines L^{VM} that need to be placed in the data center. The goal is to place VMs in L^{VM} on the minimum number of PMs in accordance with the current resource demands.

When the utilization of PM resources is low, all VMs must migrate to other PMs, and such a PM is defined to be switched to the sleep mode. In another case, when a PM is overloaded, at least one VM must be migrated to reduce the number of SLA violations. The result of Q-learning algorithm is the determination of each PM mode (sleeping or active).

The algorithm can dynamically adjust the number of PMs to the workload changes. The PMs involved in the management process are tagged as follows: (i) the tag SLEEP indicates the PM which need to be switched to the sleep mode, (ii) the tag PLACEMENT indicates the PM which is involved in the process of VM consolidation, (iii)

the tag AVAILABLE indicates the PM which is available for the VM placement process.

The pseudo-code for the algorithm of dynamic VM consolidation is presented in [5]. In this paper, only three types of resources are used (*CPU*, *RAM*, *BW*) to run simulations. Such a choice is due to the use of the extended version of CloudSim [7] to enable energy-aware simulations. Besides, the real-world workload traces from Bitbrains [28] are used as an input data set for simulations.

The number of available PMs (value q in [5]) is selected according to the management policy that takes into consideration the dynamic of VM lifetime in cloud data center. The VM lifetime can be estimated by using a coefficient of VM vitality denoted by VMV. The VMV metric is proposed by the authors in [29] and can be computed as follows:

$$VMV = \frac{N_{VM}^{on}}{N_{VM}^{off}}, \tag{3}$$

where N_{VM}^{on} is a number of new VM deployments at the previous management step, N_{VM}^{off} is a number of VMs that was turned off at the previous management step.

The VMV metric is used for decision making process. If $VMV < 1$ for the short-term control horizon then the policy that controls the number of PMs serving VMs may be corrected to perform VM consolidation process more aggressively and the value of q tends to be equal to the number of available PMs. If $VMV > 1$ for the short-term control horizon then the correspondent policy must turn on some number of PMs and the value of q is less than the number of available PMs in active mode.

An effective VM consolidation should reduce the following quality indicators: (i) the number of SLA violations, (ii) the number of active PMs, and (iii) the number of VMs migrations. The learning agent decides about when to switch PM to the sleep mode or to the active mode.

Each element of the state space *S* represents the current CPU utilization, the utilized amount of RAM, and the utilized amount of network bandwidth of each VM on each PM as follows:

$$\begin{aligned} s_t = \{&CPU_{PM1}, RAM_{PM1}, BW_{PM1}\}, \\ &\{CPU_{PM2}, RAM_{PM2}, BW_{PM2}\}, ..., \\ &\{CPU_{PMm}, RAM_{PMm}, BW_{PMm}\}, \end{aligned} \tag{4}$$

where $i = \overline{1,m}$, m is the number of PMs, $CPU \in [0,1]$ is the CPU utilization, $RAM \in [0,1]$ is the utilized amount of RAM, $BW \in [0,1]$ is the indicator of network utilization.

Each resource usage indicator is normalized relative to the maximum volume of the corresponding PM resource. To reduce the number of states, the usage rate for each resource is rounded to two decimal digits.

The action space is defined as the set $A = \{a_1(t), a_2(t), ..., a_i(t), a_m(t)\}$, $a_i(t) \in \{-1, 0, 1\}$, $i = \overline{1,m}$. Each action of *A* transfers the PM to the sleep mode or to the active mode before the next management step.

The penalty value p_t consists of two values: a penalty for SLA violations p_t^{SLA} and a penalty for power consumption p_t^{power}. The penalty value can be defined as follows:

$$p_t = \beta p_t^{SLA} + \delta p_t^{power}, \tag{5}$$

where β and δ are weights that determine relative importance of p_t^{SLA} and p_t^{power} correspondingly.

The p_t^{SLA} value is calculated by dividing the total time of SLA violations over all PMs after completing the action $a_t \in A$ for the total time of SLA violations after action in the previous management step $a_{t-1} \in A$. The penalty for SLA violations is defined as follows:

$$p_t^{SLA} = \begin{cases} \sum_{i=1}^{m} \frac{T_t^{SLA}}{T_{t-1}^{SLA}}, & T_{t-1}^{SLA} > 0 \\ 0, & T_{t-1}^{SLA} = 0 \end{cases}, \tag{6}$$

where T_t^{SLA} is a time of SLA violations after completing the action $a_t \in A$; T_{t-1}^{SLA} is a time of SLA violations after completing the action $a_{t-1} \in A$.

The penalty for power consumption can be calculated as the sum of power consumption of all PMs as follows:

$$p_t^{power} = \sum_{i=1}^{m} \frac{P_{i,t}}{P_{i,t-1}}, \tag{7}$$

where $P_{i,t}$ is the power consumption by *i*-th PM in the current management step, $P_{i,t-1}$ is the power consumption by *i*-th PM in the previous management step.

All VM placements and VM migrations as well as switching the PM mode must be completed before the beginning of the next management step *t*+1. Then, the Q-value for each state-action pair of the management step is updated through the total amount of penalty p_t.

V. Simulation Results

A. Description of the simulation environment

For the evaluation of the proposed reinforcement learning method the authors have run simulations with Bitbrains traces [8] and analyzed the logs regarding the SLA violation, the power consumption, and the number of VMs migrations during simulation time. To evaluate the proposed reinforcement learning method, the CloudSim toolkit [30] is used with custom enhancements as required. It is a modular and extensible open source toolkit which has built-in capability to implement and compare power-aware management algorithms for different cloud environments and workloads. The extended version of CloudSim is used [7] to enable power-aware simulations. The simulations were conducted on the computer with the Intel i7-3632QM processor and 8 GB of RAM running Windows 10 Pro 64bit.

TABLE I. PHYSICAL MACHINES CHARACTERISTICS

PM Type	Number	MIPS of CPU	Number of PEs	RAM capacity	Bandwidth, Gbit/s
Dell Inc. PowerEdge R640	50	2500	56	196608	1
Dell Inc. PowerEdge R740	50	2500	56	196608	1
Dell Inc. PowerEdge R830	50	2200	88	262144	1
Dell Inc. PowerEdge R940	50	2500	112	393216	1

TABLE II. VIRTUAL MACHINES CHARACTERISTICS

PM Type	Number	MIPS of CPU	Number of PEs	RAM capacity	Bandwidth, Mbit/s
Type 1	125	500	2	2048	100
Type 2	125	1300	4	4096	100
Type 3	125	2200	8	8192	100
Type 4	125	2500	16	16384	100

The simulated data center comprises 200 heterogeneous PMs with the characteristics shown in Table I. Besides, in each simulation the authors use 500 heterogeneous VMs with different numbers of cores from among 4 VM types as shown in Table II. The VM types correspond to the resources requested by the VMs reported in [8]. In all experiments, the number of VMs was not changed, but in a general case the number of VMs can vary. Moreover, in dynamic environment some VMs determined for migration may cease to exist during the management step.

The workload traces from a private cloud data center [28] were used as a workload for simulations. Initially, the set contains 1200 VMs traces for the resource usage of Web/application/database servers, where the VMs belongs to 44 different classes, with each class containing from 8 to 50 VMs [8]. Only 500 traces were used as the input in the simulations. In each trace file, the monitoring data of one VM about utilization of the processor, memory, and network interface are stored. All experiments are carried out under the dynamic workload with 500 VMs for a simulation period of 9 days in which the interval of utilization measurements is 300 seconds.

For experiments, the authors modify the CloudSim classes in such a way that three types of resources (*CPU*, *RAM*, *BW*) are taken into account during simulations. Besides, CloudSim classes were modified to account different power consumption profiles at different rates of a server processor utilization provided by the SPEC [31].

B. Estimation of the model parameters

The objective of the experimental phase has been to evaluate the quality indicators of reinforcement learning method under different workload and to compare them with quality indicators of the methods proposed in [13].

The evaluation of the effectiveness of the proposed method has been performed by considering three quality indicators: the SLA violation time, the power consumption and the number of VM migrations. At the first stage, it is necessary to study the effect of the learning rate α and the discount factor γ on the effectiveness of the proposed method. At the second stage, the authors investigate the effect of weights β and δ on the performance indicators of the proposed method. At the third stage, the authors compare the proposed method with methods and heuristics of the VM consolidation presented in [13].

The authors vary the learning rate α from 0.1 to 1 in steps of 0.25 and the discount factor γ from 0.1 to 1 in steps of 0.25. The weights β and δ were kept fixed at 0.5. As a result of 25 experiments, the values of α and γ that ensure the minimal SLA violation time were defined as follows: $\alpha = 0.5$, $\gamma = 0.5$.

At the second stage, weights β and δ were varied from 0.1 to 1 in steps of 0.25 while $\beta + \delta = 1$. The values of α and γ were kept fixed at 0.5 as defined at the first stage. As a result of 11 experiments, the values of weights β and δ that ensure the minimal SLA violation time were defined as follows: $\beta = 0.8$, $\delta = 0.2$. The values of weights β and δ that ensure the minimal power consumption were defined as follows: $\beta = 1$, $\delta = 0$.

At the third stage, the authors conduct simulations of the proposed reinforcement learning method and the methods proposed in [13] using previously defined values of α, γ, β, and δ.

Figure 2 shows cumulative values of the SLA violation indicator, the power consumption indicator, and the number of VM migrations. The authors analyze its aggregate usage over time by summing, each management step, the correspondent indicators observed for all the PMs. The results shown in Fig. 2 indicate that the proposed method outperforms the competitor methods [13] in SLA violation time and in the number of VM migrations because the proposed method performs less VM migrations and turns to the sleep mode less PMs than competitor methods do. As a result, the proposed method ensures less time of SLA violations during management process. The methods presented in [13] ensures low power consumption due to much greater number of migrations that usually is not acceptable in production [32]. Moreover, the competitor methods do not consider power consumption during PM setup and PM sleep mode. That can be serious drawback when the large number of PMs change their states frequently [23].

VI. ANALYSIS

The extensive simulations show good results for competitor methods [13] with static configuration and fixed number of PMs and VMs. But the number of VMs in

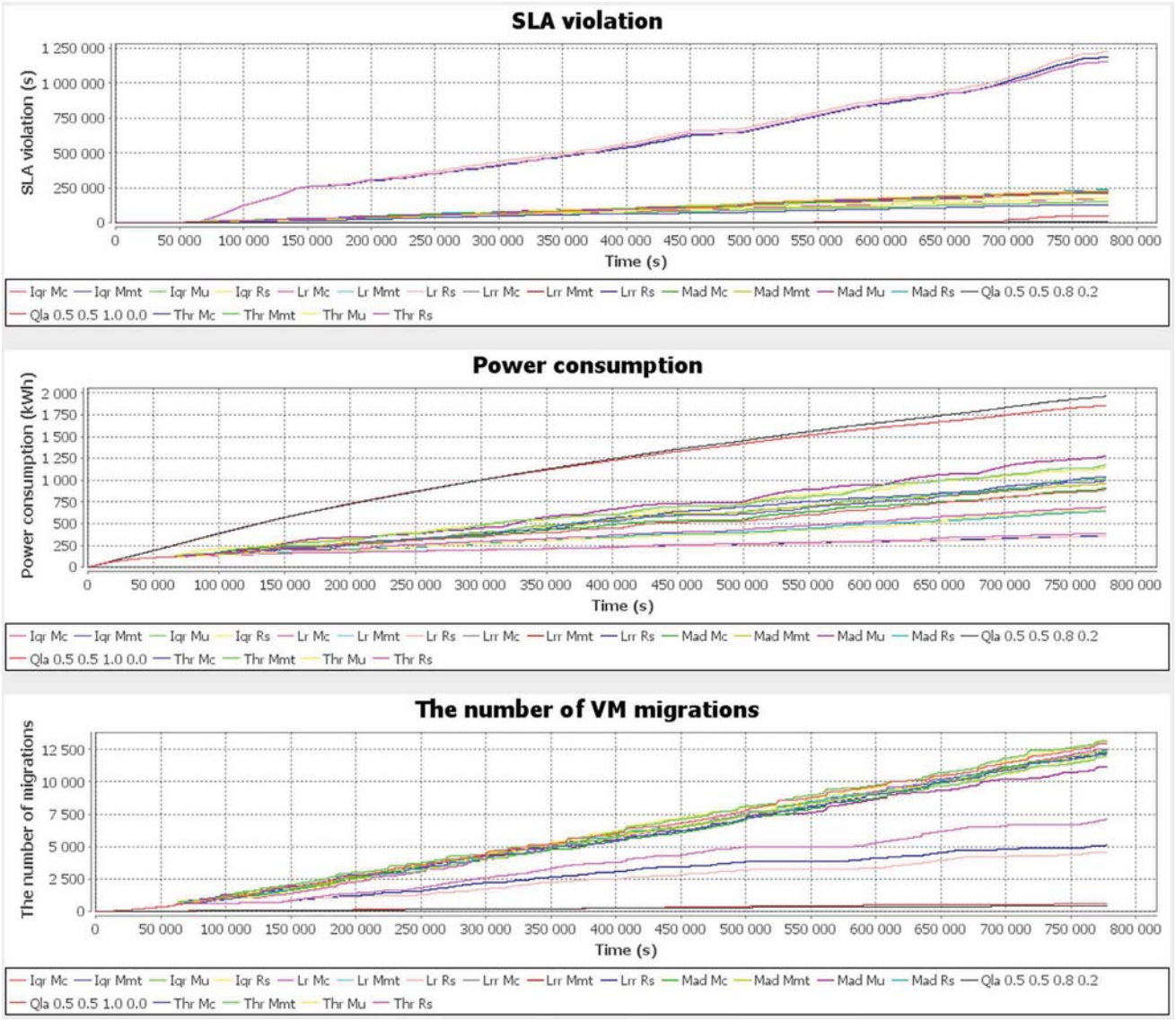

Fig. 2. Cumulative values of the quality indicators for the proposed method (Qla) and the competitor methods [13].

TABLE III. PM TRANSITION TIME [34]

Physical server model	Setup mode				
	OFF - IDLE	SLEEP - IDLE	IDLE - SLEEP	SLEEP - OFF	IDLE - OFF
HpProLiantMl110G3PentiumD930	50	20	3	2	5
HpProLiantMl110G4Xeon3040	45	10	3	2	5
HpProLiantMl110G5Xeon3075	50	20	3	2	5
IbmX3250XeonX3470	45	10	3	2	4
IbmX3250XeonX3480	45	10	2	2	4
IbmX3550XeonX5670	60	20	3	2	5
IbmX3550XeonX5675	75	30	3	2	5

production data center can change over time. Thus, there are two disadvantages of the policy [13] that perform simple power consumption minimization: (i) high VM scheduling delay due to the absence of the active PM to satisfy VM resource requirements, (ii) the increased power consumption when switching a PM from the active state to the sleep mode and vice versa (setup mode [21]) when the number of VMs changes frequently during a relatively short period of time.

On the other hand, an AlwaysOn policy [33] ensures minimal VM scheduling delay and leads to maximal power consumption. The AlwaysOn policy is a static power management policy that is widely used by cloud service providers. That policy maintains a fixed number of PMs in active (idle) state at all times. Then, an important problem

raised here is to find a balance between AlwaysOn policy and the policies proposed in [13]. But the solution of this problem is beyond the scope of this paper.

Setup mode takes significant time when a PM is switching from the sleep mode to active mode. The time spent to switch a PM between the sleep mode and the active mode is called a setup time. The higher the setup time, the higher the VM scheduling delay when there are no active PMs that can satisfy user's VM scheduling request. The setup time for some production servers [34] is presented in Table III.

The setup time of a PM varies from 20 seconds to 200 seconds depending on the hardware and software configuration and can be as large as 260 seconds [21]. The power consumption during the setup time is close to the maximal rate for the PM [22]. Similar results were reported in [19], [20]. During the setup time, the servers consume about 200W [20]. Thus, the high setup times force data center providers to implement any form of dynamic power management policies to switch specific servers to the sleep mode when the workload drops.

Another important result of the presented extensive simulations is the influence of the number of input parameters that are taken into account during CloudSim simulation. The competitor methods reported in [13] take into account only CPU workload [18] as an input to each modeled VM. In the presented simulations three types of resources are used (*CPU*, *RAM*, *BW*) as an input to each modeled VM. Thus, the reported in [13] results are different from the results obtained by the modified CloudSim toolkit in terms of the power consumption and the SLA violation time.

Thereby, the main benefits of the proposed reinforcement learning method are: the possibility to reduce the SLA violation time that can be very expensive in production, the possibility to serve more VM schedule requests when the number of VMs is changing frequently, the possibility to decrease the utilization of data center network due to decreased number of migrations.

VII. Conclusion

In this paper, the authors refine and evaluate the previously proposed method for cloud data center resource management based on the reinforcement learning approach. The proposed reinforcement learning agent allows to determine the optimal policy for managing the physical servers without creating an environment model and preliminary information about the workload.

The performance evaluation based on real workload traces from Bitbrains [28] demonstrates the effectiveness of the proposed method. The proposed method allows to decrease the SLA violation time, to serve more VM schedule requests when the number of VMs is changing frequently, and to decrease the utilization of data center network due to decreased number of migrations.

The previously proposed method [5] is refined by elaborating of the core algorithm and by decreasing the size of state space, the number of the resource utilization intervals and the number of the state space element attributes. As a part of future work, the authors plan to improve the proposed method by accounting power consumption in setup mode. Another planned improvement is the adaptation of CloudSim environment to dynamic changes of the VMs number during simulation.

References

[1] L. M. Gonzalez, Vaquero, L. Rodero-Merino, J. Cáceres, and M. Lindner, "A break in the clouds: towards a cloud definition," *Computer Communication Review*, vol. 39, pp. 50-55, 2008.

[2] M. Buco, R. Chang, L. Luan, C. Ward, J. Wolf, and P. Yu, "Utility computing SLA management based upon business objectives," *IBM Systems Journal*, vol. 43, no. 1, pp. 159-178, 2004.

[3] A. Varasteh and M. Goudarzi, "Server Consolidation Techniques in Virtualized Data Centers: A Survey," in *IEEE Systems Journal*, vol. 11, no. 2, pp. 772-783, June 2017.

[4] A. Ashraf, B. Byholm, I. Porres, "Distributed virtual machine consolidation: A systematic mapping study," *Computer Science Review*, vol. 28, pp. 118-130, 2018.

[5] O. Rolik, E. Zharikov, A. Koval and S. Telenyk, "Dynamic management of data center resources using reinforcement learning," *2018 14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET)*, Lviv-Slavske, 2018, pp. 237-244.

[6] S. Shen, V. van Beek, and A. Iosup, "Statistical characterization of business-critical workloads hosted in cloud datacenters," in *15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, 2015, pp. 465–474.

[7] F. Lopez Pires and B. Baran, "A virtual machine placement taxonomy," in Proc. of the *15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid),* 2015, pp. 159–168.

[8] D. Weerasiri, M. C. Barukh, B. Benatallah, Q. Z. Sheng, and R. Ranjan, "A taxonomy and survey of cloud resource orchestration techniques," *ACM Computing Surveys (CSUR)*, vol. 50, no. 2, p. 26, 2017.

[9] A. Yousafzai, A. Gani, R. M. Noor, M. Sookhak, H. Talebian, M. Shiraz, and M. K. Khan, "Cloud resource allocation schemes: review, taxonomy, and opportunities," *Knowledge and Information Systems*, vol. 50, no. 2, pp. 347–381, 2017.

[10] S. H. H. Madni, M. S. A. Latiff, Y. Coulibaly, and others, "Resource scheduling for infrastructure as a service (IaaS) in cloud computing: Challenges and opportunities," *Journal of Network and Computer Applications*, vol. 68, pp. 173–200, 2016.

[11] 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*, vol. 24, no. 13, pp. 1397–1420, 2012.

[12] L. Chen and H. Shen, "Considering resource demand misalignments to reduce resource over-provisioning in cloud datacenters," in *INFOCOM 2017-IEEE Conference on Computer Communications, IEEE*, 2017, pp. 1–9.

[13] H. Shen and L. Chen, "Distributed autonomous virtual resource management in datacenters using finite-markov decision process," *IEEE/ACM Transactions on Networking*, vol. 25, no. 6, pp. 3836–3849, 2017.

[14] M. A. H. Monil and A. D. Malony, "QoS-Aware Virtual Machine Consolidation in Cloud Datacenter," in *2017 IEEE International Conference on Cloud Engineering (IC2E)*, 2017, pp. 81–87.

[15] X. Zhou, K. Wang, W. Jia, and M. Guo, "Reinforcement learning-based adaptive resource management of differentiated services in geo-distributed data centers," in *2017 IEEE/ACM 25th International Symposium on Quality of Service (IWQoS)*, 2017, pp. 1–6.

[16] K. S. Park, V. S. Pai, "CoMon: a mostly-scalable monitoring system for PlanetLab," *ACM SIGOPS Operating Systems Review*, vol. 40(1), no. 1, pp. 65-74, 2006.

[17] I. Sarji, C. Ghali, A. Chehab, and A. Kayssi, "Cloudese: Energy efficiency model for cloud computing environments," in *2011 International Conference on Energy Aware Computing (ICEAC)*, 2011, pp. 1–6.

[18] S. L. Xi, M. Guevara, J. Nelson, P. Pensabene, and B. C. Lee, "Understanding the critical path in power state transition latencies,"

in *2013 IEEE International Symposium on Low Power Electronics and Design (ISLPED)*, 2013, pp. 317–322.

[19] A. Gandhi, M. Harchol-Balter, and M. A. Kozuch, “Are sleep states effective in data centers?,” in *2012 International Green Computing Conference (IGCC)*, 2012, pp. 1–10.

[20] A. Gandhi, M. Harchol-Balter, R. Raghunathan, and M.Kozuch. “AutoScale: dynamic, robust capacity management for multi-tier data centers,” *ACM Trans. on Computer Systems*, vol. 30(4), pp. 1–26, 2012.

[21] A. Paya and D. C. Marinescu, “Energy-aware load balancing and application scaling for the cloud ecosystem,” *IEEE Transactions on Cloud Computing*, vol. 5, no. 1, pp. 15–27, 2017.

[22] X. Dutreilh, A. Moreau, J. Malenfant, N. Rivierre and I. Truck, "From Data Center Resource Allocation to Control Theory and Back," *2010 IEEE 3rd International Conference on Cloud Computing*, Miami, FL, 2010, pp. 410-417.

[23] C. J. C. H. Watkins and P. Dayan, “Technical note: Q-learning,” *Machine Learning*, no. 3(8), pp. 279–292, 1992.

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

[25] J. Yang, S. Zhang, X. Wu, Y. Ran, and H. Xi, “Online Learning-Based Server Provisioning for Electricity Cost Reduction in Data Center,” *IEEE Transactions on Control Systems Technology*, vol. 25, no. 3, pp. 1044–1051, 2017.

[26] GWA-T-12 Bitbrains [Online] Available from: http://gwa.ewi.tudelft.nl/datasets/gwa-t-12-bitbrains [May 12, 2018].

[27] E. Zharikov, O. Rolik and S. Telenyk, "An integrated approach to cloud data center resource management," in *2017 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T)*, Kharkov, 2017, pp. 211-218.

[28] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, and R. Buyya, “CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms,” *Software: Practice and Experience*, vol. 41, no. 1, pp. 23–50, 2011.

[29] Standard Performance Evaluation Corporation [Online] Available from: http://spec.org/ [May 12, 2018].

[30] Limits on Simultaneous Migrations [Online] Available from: https://docs.vmware.com/en/VMware-vSphere/6.0/com.vmware.vsphere.vcenterhost.doc/GUID-25EA5833-03B5-4EDD-A167-87578B8009B3.html [May 12, 2018].

[31] Akshat Verma, Gargi Dasgupta, Tapan Kumar Nayak, Pradipta De, and Ravi Kothari, “Server workload analysis for power minimization using consolidation,” In *Proceedings of USENIX ATC 2009*, pp. 355–368.

[32] M. R. V. Kumar and S. Raghunathan, “Power management using dynamic power state transitions and dynamic voltage frequency scaling controls in virtualized server clusters,” *Turkish Journal of Electrical Engineering \& Computer Sciences*, vol. 24, no. 4, pp. 2290–2306, 2016.