Virtual Machine Placement in Cloud systems using Learning Automata

N. Rasouli[1]
Department of Electronic, Computer and Electrical Engineering,
Qazvin Islamic Azad University,
Qazvin, Iran.
n.rasouli@qiau.ac.ir[1]

M. R. Meybodi[2]
Department of Computer Engineering,
Amir Kabir University of Technology,
Tehran, Iran.
mmeybodi@aut.ac.ir[2]

H. Morshedlou[3]
Department of Computer Engineering,
Amir Kabir University of Technology,
Tehran, Iran.
morshedlou@aut.ac.ir[3]

***Abstract* — In recent years, the IT infrastructure due to the demand for computing power which used by applications are rapidly growing and modern data centers in cloud computing are hosting a variety of advanced applications. The high energy cost and green-house gas emissions are significant problems that have emerged as results of using large data centers. Thus providing an efficient method to reduce energy consumption by data centers is highly regarded by researchers. In this paper we present a new approach based on Learning Automata for dynamic replacement of virtual machines over data centers to reduce power consumption. Live migration and forcing idle nodes to sleep constitute main policies of this approach. To evaluate the proposed method, the workload is used in the real world. Simulation results show that the performance of the proposed method significantly reduces the energy consumption However, the efficiency of the system is preserved to a considerable extent.**

Keywords: energy efficiency; learning automata; resource allocation; cloud computing.

I. Introduction

Modern data centers which used in the cloud computing are hosting various kinds of applications. It is possible for only a few seconds or for longer periods of time are running in data centers. In the past, the main criterion in the design of data centers has been high performance and power consumption of these centers has been of little importance. But today, as the importance of energy in the world, energy consumption is regarded as one of the most important criteria.The average energy consumption of data centers are equivalent to 25,000 households [1]. In addition to the operating costs, high energy consumption leads to an increase in temperature will reduce the reliability and longevity of the hardware resources. Because it produces substantially carbon dioxide (CO2) emissions contributing to the greenhouse effect so as environmental aspects are taken into consideration [2]. Energy consumed by a computing node in a data center includes the processor, storage and networking equipment and stored resources. Compared to other system resources, CPU will consume large amounts of energy [1].

Due to the presence of energy dissipation in a passive server substantial amount of energy is consumed. Providers must be able to provide a quality service to meet the energy efficiency. So one of the important requirements for a cloud computing environment have to provide reliable quality of service which can be defined in rules "service level agreement". In fact, "service level agreement" explains features such as minimum throughput, maximum response time, or delay in delivery by the deployed system. Using power-aware scheduling mechanism can be significant savings in energy consumption, and these mechanisms must be a compromise between efficiency and reduction services cost for savings energy. One way to reduce energy consumption by data centers is virtualization approach. In a general definition, virtualization means create an abstract level of computing resources that can be applied at different levels of a machine. Hardware level, platforms and applications are some case that can create abstraction about them. Virtualization improves productivity by making efficient use of their resources and can reduce the amount of hardware resources needed. Because virtual machines seat on a hardware resource are independent from each other so finish time of their activity is different from each other and this make the part of capacity on hardware machines be unused. A new technique has been developed to address this problem, is migration. By using this technique with moving and finding a better place for each virtual machine can have data centers with higher performance and the other way can reduce the lateral cost among energy cost. Migration as one of the achievements of virtualization, today by most data centers are used for various purposes such as load balancing, building servers, preventing downtime, etc.

Allocation of Energy-aware data center resources to support the relocation of virtual machines between the physical nodes, enables dynamic migration of virtual machines according to the performance requirements. When virtual machines do not use all resource provided, they can be modified to a reasonable size and deployed to the minimum number of physical nodes. To reduce energy consumption by the idle node and also data center total energy consumption, can change those nodes to sleep mode or turn them off [2]. One other solutions is the methods related to energy management that includes on / off system components, sleeping methods, etc[3]. A lot of research has been done in data centers virtual machine placement. These approaches include the reduction of energy consumption [4] [5], [6], the cost allocation, nodes thermal in data centers [7], the network communication paths between virtual machines [8], [9]. Using genetic algorithms for solving approach, combined energy reduction with network communication [10] have been proposed. Optimization over multiple system resources – at each time frame VMs are reallocated according to current CPU, RAM and network bandwidth utilization. In this paper we present a new approach based on Learning Automata for dynamic replacement of virtual machines over data centers to reduce power consumption. Live migration and forcing idle nodes to sleep constitute main policies of this approach .proposed approach unlike past algorithms does not use previous solutions but tries to use the ability of learning automata do virtual machine placement.The proposed approach can effectively handle heterogeneous infrastructure and heterogeneous virtual machines and do not require any knowledge about applications executing on virtual machines.

II. Learning automata

A. *Finite Action-Set Learning Automata*

Learning automata is an abstract model that chooses an action from a finite set of its actions randomly and takes it. In this case, environment evaluates this taken action and reacts to it by a reinforcement signal. Then, learning automata updates its internal information regarding both the taken action and received reinforcement signal. After that, learning automata chooses another action again. Figure 1 depicts the relationship between learning automata
and environment.

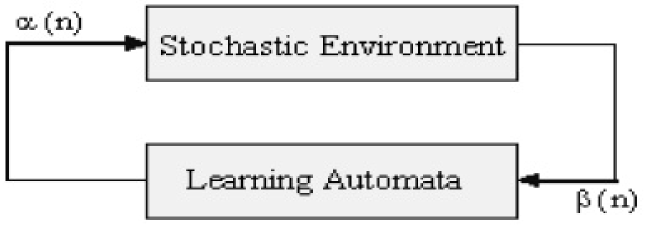

Fig. 1. Interaction between learning automata and environment

Every environment is represented by $E = \{\alpha, \beta, c\}$,where $\alpha = \{\alpha_1\ \alpha_2, \ldots, \alpha_r\}$ is a set of inputs, $\beta = \{\beta_1, \beta_2, \ldots, \beta_r\}$is a set of outputs, and $c = \{c_1, c_2, \ldots, c_r\}$ is a set of penalty probabilities. Whenever set β has just two members, environment is kind of *p*. In this environment $\beta_1 = 1, \beta_2 = 0$ are considered as penalty and reward respectively. Similarly, environment in kind of *Q* contains a finite set of members. Also, environment in kind of *S* has infinite number of members. c_i is the penalty probability of taken action α_i.Learning automata is classified into fixed structure and variable structure. Learning automata with variable structure is introduced as follows; Learning automata with variable structure is represented by $\{\ \alpha, \beta,\ p, T\}$, where $\alpha = \{\alpha_1\ \alpha_2, \ldots, \alpha_r\}$ is a set of actions, $\beta = \{\beta_1, \beta_2, \ldots, \beta_r\}$is a set of inputs, $p = \{p_1, p_2, \ldots, p_r\}$ is the action probability vector, and $p(n+1) = T[\ \alpha(n), \beta(n), p(n)]$ is learning algorithm. Learning automata operates as follows; learning automata chooses an action from its probability vector randomly (*Pi*) and takes it. Suppose that the chosen action is α_i. Learning automata after receiving reinforcement signal from environment updates its action probability vector according to formulas 1 and 2 in case of desirable and undesirable received signals respectively. In formulas 1 and 2, a and b are reward and penalty parameters respectively. If $a = b$ then algorithm is named L_{R-P} . Also, if $b<<a$ then the algorithm is named $L_{R\varepsilon P}$. Similarly, if $b = 0$ then the algorithm is called L_{R-I} .

(1) $$p_i(n+1) = p_i(n) + a.\left(1 - p_i(n)\right)$$
$$p_j(n+1) = p_j(n) + a.p_j(n)$$

(2) $$p_i(n) = (1-b).\left(p_i(n)\right)$$
$$p_j(n+1) = \frac{b}{r-1} + (1-b)p_j(n)$$

B. *Continuous Action-Set Learning Automata (CALA)*

So far, we have considered the LA model where the set of actions is finite. Hence, while finding the optimal parameter values to maximize a performance index, we need to discretize the parameter space so that actions of LA can be possible values of parameters. A more satisfying solution would be to employ an LA model where the action-set can be continuous. Such a model is called *continuous action-set learning automaton or CALA.* The action-set of CALA is the real line. The action probability distribution at k is $\mathcal{N}(\mu(k), \sigma(k))$, the normal distribution with mean $\mu(k)$ and standard deviation$\sigma(k)$. At each instant, the CALA updates its action probability distribution by updating $\mu(k)$ and $\sigma(k)$. Let $\alpha(k) \in \Re$ be the action chosen at k and let $\beta(k)$ be the reinforcement at k. Here, instead of reward probabilities for various actions, we now have a reward function $\mathcal{F}: \Re \to \Re$ defined by $f(x) = E\ [\beta(k)|\alpha(k) = x]$. We shall denote the reinforcement in response to action as and thus $f(x) = E\beta_x$. The objective for CALA is to learn the value of x at which attains a maximum$f$. That is, we want the action probability distribution $\mathcal{N}(\mu(k), \sigma(k))$ to converge to $N(x_0, 0)$ where $\boldsymbol{x_0}$ is a maximum of . However, we do not let $\sigma(k)$ converge to zero to ensure that the algorithm does not get stuck at a nonoptimal point. Therefore, we use another parameter, $0 < \sigma_\ell$ (with $\boldsymbol{\sigma_\ell}$ sufficiently small) and keep the objective of learning as $\sigma(k)$ converging to $\boldsymbol{\sigma_\ell}$ and $\mu(k)$ converging to a maximum of f.
The learning algorithm for CALA is described next. Since the updating given for $\sigma(k)$ does not automatically guarantee that $\sigma(k) > \sigma_\ell$, we always use a projected version of $\sigma(k)$, denoted by $\emptyset(\ \sigma(k))$, while choosing actions. Furthermore, unlike FALA, CALA interacts with the environment through a choice of two actions at each instant.
At each instant k, CALA chooses a $x(k) \in \Re$ at random from its current action probability distribution $\mathcal{N}(\mu(k), \emptyset(\ \sigma(k)))$, where $\emptyset$ is the function specified below. Then, it gets the reinforcement from the environment for the two actions: $\mu(k)$ and $x(k)$. Let these reinforcements be β_μ and β_x . Then, the distribution is updated as follows:

$$\mu(k+1) = \mu(\mathrm{k}) + \lambda \frac{(\beta_x - \beta_\mu)}{\emptyset(\sigma(k))} \frac{(x(\mathrm{k}) - \mu(\mathrm{k}))}{\emptyset(\sigma(k))}$$
(1)

$$\sigma(k+1) = \sigma(\mathrm{k}) + \lambda \frac{(\beta_x - \beta_\mu)}{\emptyset(\sigma(k))} \left[\left(\frac{(x(\mathrm{k}) - \mu(\mathrm{k}))}{\emptyset(\sigma(k))} \right)^2 - 1 \right] + \lambda\{\mathcal{C}[\sigma_\ell - \sigma(\mathrm{k})]\}$$
(2)

III. PROPOSED APPROACH (PBLA)

Placement algorithms have common steps that in the following four steps can be briefly stated. How they do these steps are different based on their proposed approach:

Step 1: Select the sender host.
Step 2: Select the virtual machines to migrate.
Step 3: Select the receiving host.
Step 4: Allocate the virtual machine.

It seems that, according to the nature of random availability of physical nodes in the cloud's platform and access to distribution feature, the local Community, using of distributed reinforcement learning model to be helpful. One of the reinforcement learning tools is stochastic learning automata which in this paper used, in order to learning. As described in the previous section, the random automaton without any information about the optimal action (means considering equal probability for all its actions at the beginning) is trying to answer to the problem: as a component of network of learning automata with a finite number of actions in which each automata's decision is independent of the others. B) For learning the optimal value of a continuous parameter in which a certain type of learning automata is applied to, set of its action is a continuous intervals. In further each of these areas will be explored more. Figure 3 gives the pseudo code for the PBLA approach.

(A) Any physical nodes in the cloud's data centers, is an automata that its learning is accept or reject the virtual machine from the source node to the destination node. Each learning automata has two actions which only one of them active at a time. According to the utilization of processors, we have defined some status. Four defined status explain as follows:

- Idle status ($\boldsymbol{S_{Idle}}$), CPU usage at intervals (0.0, 0.1], the nodes must be turn off for saving energy and prevent the allocation of the virtual machine to it.
- Average status ($\boldsymbol{S_{\mathrm{Average}}}$), CPU usage at intervals (0.1, 0.6), this node has fairly average workload.
- Active status ($\boldsymbol{S_{\mathrm{Active}}}$), the CPU usage in the range [0.6, 0.8), the server is active and it is in the set of active nodes.
- Over utilized status ($\boldsymbol{S_{Over}}$), the CPU usage in the range [0.8, 1)

Those hosts which their CPU utilization is equal to zero do not mention to above category and consider them as a shut-down hosts.

From a mathematical point of view, each of this status is equivalent to the state of a Markov chain [12]. Transition from one state to another state is equivalent to change state of a node to another state after virtual machine admission. Figure 2 shows the general form of states change. Each learning automata has its own Markov chain and at each of these states of chain are doing their learning independently from other states. Because of the different benefit of each action in different case, without being specific case, automata would not be capable of learning. For proper management of virtual machines dynamic migration in data center and also prevention of this problem, which a host can be both transmitter and receiver at the same time; we have categories behavior of the host as the sender host, receiver host, and neutral. Each host no matter is in which category at each instant based on the itself present state can be located in a different state or remain in the previous state.

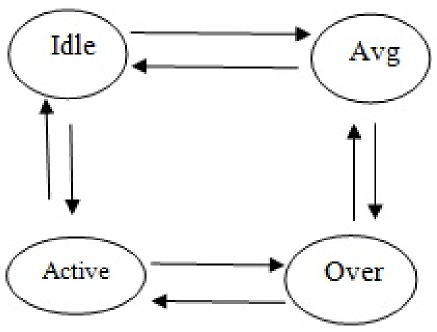

Fig. 2. Automaton state changes

Learning automata coefficients of reward and penalty whose values will be determined according to the source and destination nodes. Nodes of destination use this environment response, in the next iteration until the new assignment is created to be closer to the optimal schedule. This iterations recurs great number until find appropriate placement and do not change in their decision. In fact, iterations continue until that all automata be convergent and identical allocation be created in recur. After that, Finally every host which accepts any virtual machine, that virtual machine migrates to that host. in every time frame by considering current allocations, account all nodes energies and the whole data center energy and if recent allocations make system energy increase so allocations must modify again.

(B) For learning a parameter, we use learning automata with continuous action-set and by using this automata estimate the probability of finding receiver host for all virtual machines seat on an idle node. If this probability be higher than the specific value, machines on that host select for migration, otherwise regardless of machine migration on that host. Environment response (β) considered as a binary value, one has been assumed as a result of favorable and zero as an unfavorable responses.

```
//Placement VM using LA
Input assignment of VMs to PMs ,virtual machine requirements,
physical machine capacities, state of automata
Output re-assignment of VMs to PMs

for each $LA_i$ do
initilize the actions' probability equally
end for
        check each $LA_i$ State
choice randomly an
action for requested VM by using each state's probability
                              end if
for each idles get VMs to migrate
if CALA probability > specific val do
get VMs to migrate from host
else do not let to migrate

Finding new placement
if  (Capacity of Physical machine available) &&
(utilization of host != 0) && !(VM in migrate or out) do
     if action of $LA_i$ == accept do
          allocate $VM_j$ to $PM_i$
       check the situation of source and destination
          if $PM_j$ is active node
                         reward action of LA
else
penalize action of LA

end else
end if
end if
     if action of $LA_i$ == reject do
do not allocate $VM_j$ to PM
continue for another host to place this $VM_j$
check the situation of source and destination
          if $PM_j$ is idle node
                         reward action of LA

                           switch off the node

       else
  penalize
          action of LA

end else
end if
end if
            end
```

Fig. 3 Pseudo code for the SALA algorithm

III. Evaluation

In the process of simulation, CloudSim[13] simulator has been used. For an appropriate experiment, in simulations workload of a real system have been used. Data related to this workload have taken from the CoMon[14] project, a monitoring infrastructure for PlanetLab. We have simulated a data center that comprises 800 heterogeneous physical nodes. Users submit requests for provisioning of 1059 heterogeneous VMs that fill the full capacity of the data center.

Two policies for choosing VMs that have to be migrated from the medium and over utilization host: (a) Minimization of Migrations (MM) – migrate the least number of VMs to minimize migration overhead. (b) Random Selection Policy (RS) – migrating the necessary number of VMs by picking them according to a uniformly distributed random variable.
The proposed method in this paper, compare with the virtual machines allocation approaches. We have combined Implementation approach for better accountability and choice appropriate virtual machine with lower overhead for migration policy, virtual machine selection policy with two policies which have described.

a) DVFS that adjusts the voltage and frequency of CPU according to current utilization.
b) Single Threshold Policy (ST) the threshold policy, CPU usage is below the threshold used to maintain a constant value.
c) Interquartile Range Policy (IQR) policies that use a threshold value of the interquartile range is adaptive.

The results of these simulations are shown in Table 1. According to the results, the proposed method compared with other policies, reduce energy consumption significantly and with regard to reducing energy consumption should pay the cost of violation of service level agreements. Breach of agreement, the proposed method is higher than other algorithm, the proposed method is higher than other policies but it can be used in systems that the service level agreement is not critical there.

Table 1 Simulation Results

Policy	**Energy** KWh	**Shutdown**	**Migration**	**SLA-Vio**
DVFS	٪803.19	457	–	–
ST	٪130.12	1296	24279	11.88
IQR	٪125.52	1278	26515	11.77
PBLA	٪25.68	803	1418	14.69

Since migration is a costly process, reduce the number of migration beside more shutdown of physical nodes has caused the proposed method has better performance and is more convenient than other methods. The results are shown in Figure 4.

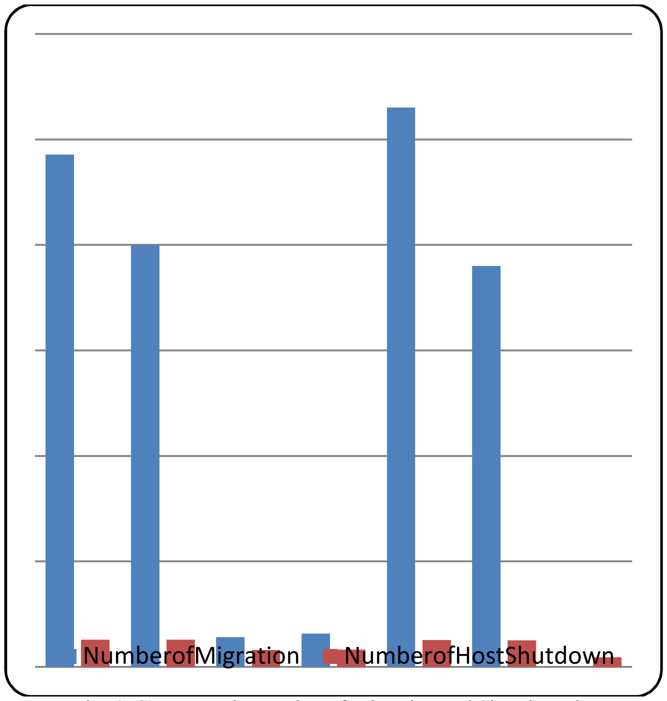

Fig. 4. Compares the number of Migration and Shut down host

Comparing the energy consumption of the different approaches is shown in Figure 5.

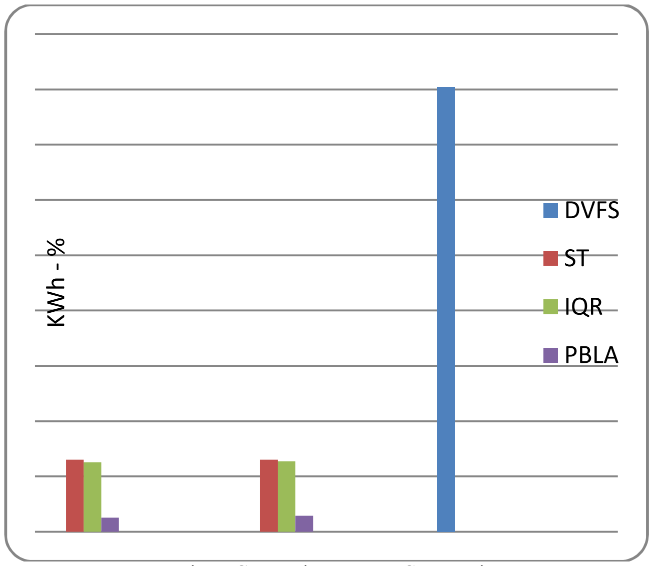

Fig.5. Comparing Energy Consumtion

III. Conclusion

In this paper have presented a new approach based on Learning Automata for dynamic replacement of virtual machines over data centers by using live migration and forcing idle nodes to shut down. It could significantly reduce energy consumption while maintaining service quality and thus reduce the heat and greenhouse gases. The proposed approach can effectively handle heterogeneous systems and according to the nature of random availability of physical nodes in the cloud's platform and access to distribution feature, the local Community, using of distributed reinforcement learning model to be helpful.

References

[1] R. Buyya, A Beloglazov " Energy-Efficient Management of Data Center Resources for Cloud Computing: A Vision, Architectural Elements, and Open Challenges" Arxiv preprint Proceedings of the 2010 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA), Las Vegas, USA, July 12-15, 2010.

[2] A.Beloglazov, R.Buyya, "Energy Efficient Resource Management in Virtualized Cloud Data Centers" 10th IEEE/ACM International Conference, 2010.

[3] L. Liu, H. Wang, "GreenCloud: A New Architecture for Green Data Center" ACM 978-1-60558-612, 2009.

[4] C.Dupont, G.Guliani "An Energy Aware Framework for Virtual Machine Placement in Cloud Federated Data Center" ACM 978-1-4503-1055-0/12/05, 2012.

[5]R.Jansen. P.Bernner "Energy Efficient Virtual Machine Allocation in the Cloud An Analysis of Cloud Allocation Policies" IEEE 978-1-4577-1221-0/11/© ,2011.

[6] Wu, Yongqiang, "A simulated annealing algorithm for energy efficient virtual machine placement" In IEEE International Conference on Systems, Man, Cybernetics, 2012.

[7]Wu, Yongqiang. "Energy-efficient virtual machine placement in data centers by genetic algorithm" In Lecture Notes on Computer Science, Springer, Renaissance Doha City Center Hotel, Doha, pp. 315-323, 2012.

[8] JT Piao, J Yan "A network-aware virtual machine placement and migration approach in cloud computing" Grid and Cooperative Computing (GCC- ieeexplore.ieee.org, 2010.

[9] X Meng, V.Pappas, L.Zhang IBM T.J. Watson Research Center "Improving the Scalability of Data Center Networks with Traffic-aware Virtual Machine Placement" Communications Society subject matter experts for publication in the IEEE INFOCOM ,2010.

[10] E. Lee, H.Viswanathan, and D. Pompili "Proactive Thermal-aware Virtual Machine Allocation in HPC Cloud Datacenters" NSF Award No. CSR-1117263. 2012.

[11] 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 (CCPE), Volume 24, Issue 13, Pages: 1397-1420, John Wiley & Sons, Ltd, New York, USA, 2012.

[12] R.Harmon,P.Challenor "A Markov chain Monte Carlo method for estimation and assimilation into models" Ecological Modelling 101, ELSEVIER page 41-59, 1997.

[13] R. Buyya, R.Ranjan, R. N. Calheiros. "Modeling and Simulation of Scalable Cloud Computing Environments and the CloudSim Toolkit: Challenges and Opportunities". pp.1-8, 2008

[14] Park KS, Pai VS. CoMon: a mostly-scalable monitoring system for PlanetLab. ACM SIGOPS Operating Systems Review 2006; 40(1):74.