International Journal of Hybrid Information Technology
Vol.8, No.7 (2015), pp.389-398
http://dx.doi.org/10.14257/ijhit.2015.8.7.35

A Cost-AWARE Approach Based ON Learning Automata FOR Resource Auto-Scaling IN Cloud Computing Environment

Khosro Mogoui and Mostafa Ghobaei Arani

Department of Computer Engineering, Islamic Azad University of Mahallat, Mahallat, Iran
Department of Computer Engineering, Islamic Azad University of Parand, Tehran, Iran
Email: khosro_mogouie@yahoo.com, mostafaghobaei@piau.ac.ir

Abstract

In recent years, applications of cloud services have been increasingly expanded. Cloud services, are distributed infrastructures which develop the communication and services. Auto scaling is one of the most important features of cloud services which dedicates and retakes the allocated dynamic resource in proportion to the volume of requests. The Scaling tries to utilize maximum power of the available resources also to use idle resources, in order to maximize the efficiency or shutdown unnecessary resources to reduce the cost of running requests. In this paper, we have suggested an approach based on learning automata for resource auto-scaling, in order to manage and optimize factor cost. Results of simulation show that proposed approach has been able to optimize cost compared to the other approaches.

Keywords: *Cloud computing, Scalability, Auto-scaling, Learning automata*

1. Introduction

Cloud computing is a new technology that has found its place in human life as a basic need, and its popularity increases among Internet users day by day. Cloud service users only pay the price based on the amount of resources they have used (Pay-per-use). Scalability is one of the important challenges and characteristics of cloud computing technology that can provide this feature for cloud services users [1-3].

One of the management techniques for scaling in the cloud computing is using a threshold approach. Threshold is determined based on factors such as the strength and speed of processing, and storage capacity. When utilization of resource is greater than the value of threshold, requests will be referred to other sources and when use of resource is less than threshold, some of the unnecessary resources will be selected and disabled to decrease costs. Therefore, the amount of implementation activities on a resource has measured periodically then the scaling will done on its basis [4].

Scalability which means allocation or withdrawal resources in proportion of requests, tries to maximize efficiency of available resources and using inactive resources in order to increase productivity or disabling available resources according to low volume of requests in order to decrease cost of performing requests. Auto- scaling enables users to make larger or smaller infrastructures according to volume of activities, desirable efficiency and other dynamic behaviors [5]. Such automation increases advantages of cloud dynamic scalability substantially and can use more resources actively at the time of high workload also it can manage cost of using resources by disabling unnecessary resources at time of low requests. Efficiency index in cloud auto-scaling mechanisms is included the amount of using CPU, disk operations and bandwidth. Also we need accurate plan to have trade-off among these factors [6].

In this paper, in addition to use previous researches results in determining threshold [7-11] and volume of workload [12], we have tried to offer an auto-scaling approach by learning automata [13], in order to management cost of scalability. Learning automata is able to select the best response among many received responses from environment. Using learning automata causes to have a better management optimal cost.

The rest of paper is organized as follows: second part is dedicated to related works. In third part, we will explain our proposed approach and in forth part we will evaluate it. Finally the last part is dedicated to conclusion and future work.

2. Related Works

Various researches have been accomplished in the field of resource auto-scaling. The aim of some of them is to offer a scalability approach for a special application like server of web and others have done for optimizing the mechanism of scalability. In this part we are going to consider some of the previous researches about auto-scaling.

M. Humphrey *et al.* [6], considered auto scaling in clouds in order to dedication fast resources with minimum cost for group of independent works. Results showed efficiency of the approach in fast resource dedication but it is not recommended for situation of unequal priority and if there is needed a special level of productivity.

Menasce *et al.* [14], presented a structure for scheduling tasks with deadlines in a heterogeneous environment. In this project, a working set (DAG), is received as input, and it is assumed that there are available a variety of services for any of sub-jobs. Decision making on the timing of the sub-jobs is done by considering its desired performance and access cost. Timing process is done by using genetic algorithm.

N. Chohan *et al.* [15], offered sample stabilization mechanism for accelerating the implementation of tasks in the context of cloud, to reduce their expenses. Overall, this research [16], has been done with the aim of providing a cost-effective structure for cloud services (profitability for cloud service provider) within the Service Level Agreement (SLA). Note that from the perspective of the customer, cost comparison is done between costs of implementation in the context of cloud and the implementation cost in the normal situation.

Roy Nilabja *et al.* [17], have proposed an algorithm which does not use reaction approach in order to auto- scaling web resources. Rather, they have suggested a predictor solution by applying the concepts proposed by Sharif *et al.* [18.19], which can be used by systems that show a mixed behavior (continuous dynamic or discrete) and it has a large set with controlling limits. The base of approach is controlling principles of volume in the future that called advanced optimization. Optimal situation is calculated through repetition in a determined period of time considering current and future limits.

Trieu C. Chiue *et al.* [20], offered an approach to auto-scale web applications in a virtual cloud environment. The most important issue in web applications is inability to design or even to predict the number of active users. The mechanism includes a first-end load balancer, and a sub-system (to keep recording) and an intelligent control system (scaling index). Load balancer is utilizing for tracking and balance for users' requests in order to access to web services on the web servers in cloud VMs. Scaling is done according to number of active users. Results show that algorithm is able to manage high traffic meeting factors of cost and good efficiency.

Jing Qi Yang *et al.* [21], offered cloud architecture for the purpose of auto-scaling resources based on anticipated workload. This method consists of two sub division pre-scaled and real time scaling. Also, the linear regression model has been used to predict traffic scaling is classified in three ways: self-healing scaling, scaling of source level and scaling of virtual machine level. The main idea in self-healing scaling is that two VMs will be able to operate overlapping. The purpose of anticipating the volume of work is estimation the number of service requests in a period of time. Deviation management between estimating traffic and real amount of traffic is the main issue in this approach.

3. The Proposed Approach

An efficient Scalable mechanism is able to meet the desired quality of service, also optimal cost for users, On the other hand, load balancer using appropriate distribution in system samples will be able to reduce the frequency of the system needed to be scaled up as much as possible to preserve stability of system. As mentioned before, the purpose in this paper is to present an approach to optimize cost of scalability by use of learning automata.

Therefore, no matter what we discuss the factors of efficiency and stability, but note it is essential that an effective approach to scaling, you should be able to establish a compromise between all the factors mentioned. Because the realization of optimum quality of service, we need to pay more. On the other hand, a minimum of cost, may be SLA Violation event

Following in this section firstly then briefly introduce the automata and finally we present our proposed algorithm.

3.1 Learning Automata

Learning algorithms struggle to improve their performance and features toward a special aim by knowing related environmental situations. [22,23]. Learning automata is kind of such algorithms which try to change its conditions probabilities, based upon responses come from surrounding then it shows special reaction in any conditions. Learning automata [24] is an abstract thing with limited actions. A learning automata's performance is in a manner that it selects one of its actions among set of its actions and applies it to the environment. Then the mentioned action will be evaluated by a random environment and automata select its next action based on response of environment. The method which automata use it for selecting next action will be determined according to the used learning algorithm. Along the process automata learns how to select optimal action. Environment includes all internal and external conditions which affect automata. Generally it is possible to present environment as a set:

$E \equiv \{\alpha, \beta, c\}$ in which α is set of inputs, β is set of outputs and c is environment penalties. Figure 1, shows relationship between learning automata and environment.

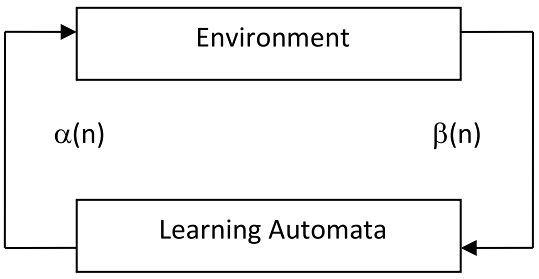

Figure 1. The Relationship Between Learning Automata And Enviroment

α is the number of actions automata is able to do, β is output of environment which differs according to the model and kind of environment Various models defined for probable environments are divided to 3 groups: P-Model, Q-Model and S-Model. In Q-model, environment outputs are discrete values of zero and one and in S-model output is always a constant value between zero and one. In this paper, we use P-Model. In this model β might be one of the two values of zero or one. Zero means desirable action and the probability of an action increases while probability of other actions decreases. One means undesirable action and probability of current action decreases and probability of other actions increase so that after receiving amount of one from environment, automata change the action. C is the set of probability of penalties for actions of automata which defines as follow:

$$c_i = \Pr ob[\beta(i) = 1 \mid \alpha(i) = \alpha_i], i = 1, 2, ..., r \tag{1}$$

These values change over the time. Values of ci often are not clear and knowing them means thorough understanding of the potential environment that is not possible in most applications. Learning automata tries to know these values. In our sample environment will operate n times in any action:

- New input load (a) enters in environment.
- Includes reward and penalty comes from environment.

Probability vector is as P={0.5, 0.5}.

According to automata operation, higher penalty in non-optimized sample, lower the chance of its selection so it can assure us of selection an optimum sample

If n repetition selects a_i so that we have in (n +1) repetition:

Received a favorable response:

$$p_i(n+1) = p_i(n) + a[1 - p_i(n)] \tag{2}$$

$$p_j(n+1) = (1-a)p_j(n) \quad \forall j \quad j \neq i \tag{3}$$

Received an unfavorable response:

$$p_i(n+1) = (1-b)p_i(n) \tag{4}$$

$$p_j(n+1) = \frac{b}{r-1} + (1-b)p_j(n) \quad \forall j \quad j \neq i \tag{5}$$

3.2 The Proposed Algorithm

Proposed algorithm includes 3 sub-algorithms. First algorithm is responsible to manage scalability process. The process includes monitoring and measuring the system efficiency in order to perform scalability. Second algorithm is responsible to select optimum samples then introducing it to the scalable management at the time of activation a new sample (high scale). Finally, the third algorithm is responsible to select suitable sample at the time of deactivation operating samples (low scale) in order to saving cost of scalability.

Generally, the process works as an algorithm (Algorithm.1), measures efficiency of active virtual machines at the time of starting any period of time and periodically at the time of working then compares them with high and low thresholds. If the measured value is higher than threshold, algorithm calls scale up to increase VMs. Otherwise, if the measured value is lower than the threshold, algorithm will call scale down to decrease VMs. If there is none of these situations, cloud will stay in a stable condition.

Algorithm 1: Scalability management

```
Begin
While (the system is running and in the beginning of
an interval)
For (every $S_{i,j}$ in the cloud)
Monitor $u_{i,j}(t)$
If ($u_{i,j}(t) > u_{upper-threshold}$)
Execute VM-level cost-aware scaling up
```

```
If ($u_{i,j}(t) < u_{lower-threshold}$)
Execute VM-level cost-aware scaling down
End
```

In the Scale up, firstly rate of SLA Violation is calculated because the amount of service requests has been increased more than the service capacity. Service capacity is specified according to the MIPS processor virtual machines of that service. Then requested amount of mentioned service–faced with lack of resources, will be calculated to meet the requests of service adding more virtual machines (Algorithm.2).

Algorithm2: Cost-aware scaling up at *t* th interval

```
Begin
Calculate SLA Violation
Calculate shortage Request
While (shortage Request is not empty)
For (every VM in VM List)
If (VM is suitable for shortage Request)
Give reward to VM based on its cost
// cheapest VM has most reward
else
Give penalty to VM based on its cost
// expensive VM has most penalty
Calculate chance(reward, penalty)
Cheapest VM← Select VM with biggest chance
// biggest chance is equal to cheapest VM
Add Cheapest VM
End
```

When the capacity of cloud service is more than issued requests of users, scalable management will issue allowance for scale down to shut down additional VMs in order to save costs of scaling (Algorithm.3). Our criteria for determining the on removal of virtual machines is that servers still have the ability to respond to users requests after deleting the virtual machine.

Algorithm 3: Cost-aware scaling down at t th interval

```
Begin
Calculate extra Request
While (extra Request is not empty)
    For (every VM in $S_{i,j}$ )
        If (VM can be removed)
            Give penalty to VM based on its cost
// expensive VM has most penalty
        Else
            Give reward to VM based on its cost
// cheapest VM has most reward
        Calculate chance(reward, penalty)
Expensive VM← Select VM with biggest chance
// biggest chance is equal to expensive VM
    Remove Expensive VM
End
```

4. Performance Evaluation

In this article, we discuss the evaluation of the proposed approach based on cost. We have used Cloudsim [25] to simulate the proposed approach. We will use 4 Types of VMs (Table 1) based on Amazon company VMs in simulation tests. Regarding variety in the existed services in cloud, we have performed four various kinds of services and we have not focused on a special service or program so that services are independent of program. The service combines all heterogeneous applications such as HPC, Web and more. The workload of the system has been modeled based on the normal distribution to make it closer to the real world.

Table 1. VM's Information

VM type	MIPS (CPU)	C ore	RAM (MB)	Price (cent)
Micro	500	1	633	0.026
Small	1,000	1	1,700	0.070
Extra Large	2,000	1	3,750	0.280
High-CPU Medium	2,500	1	850	0.560

Scaling is operated in a 24-hour period (288 * 5-minute intervals) on four different services. We considered low threshold value of 0.2 and high threshold value as 0.8 in the simulation which have been used for comparing efficiency of active VMs in system. Also, we use the probability P_i of an initial value of 0.5 and the probability P_j initial value 0.5 in order to determine the level of chance, and penalty for samples of cloud. We have tried in selection periods of time to prevent from operational overhead and scarcity so that we have considered an average period of time.

We have compared our proposed approach to the two approaches, cost-aware scaling [26] and random scaling virtual machines in the cloud computing environment to evaluate our approach. Cost-aware policy is a simple and non-learning method which decreasing cost of services is its scaling priority. Also in the random policy, there is no management on virtual machines expenses. In other words, in the Scale down and Scale up operations, adding or decreasing VMs is done randomly. We evaluate our proposed approach based on cost.

Cost: The estimated cost of the cloud service is based on the amount of working hours. The users (cloud user), depending on the speed, power and capacity of demanded resource (CPU, memory, or disk...) and also the period of acquisition (minimum of acquisition time is one hour) of resource should pay the cost. Naturally, our costs will be lower when we use the resource with lower speed and capacity, in shorter time. Perhaps this attitude optimizes your costs, but it should be noted that other factors of quality of service, will be affected. Therefore, in order to achieve high quality service, we have to bear increasing costs.

To evaluate the proposed approach, we have three policy scaling in a cloud environment, based on the assumptions (Table 1) available, the same conditions on the basis of the cost compared scaling..

4.1 Evaluation Results

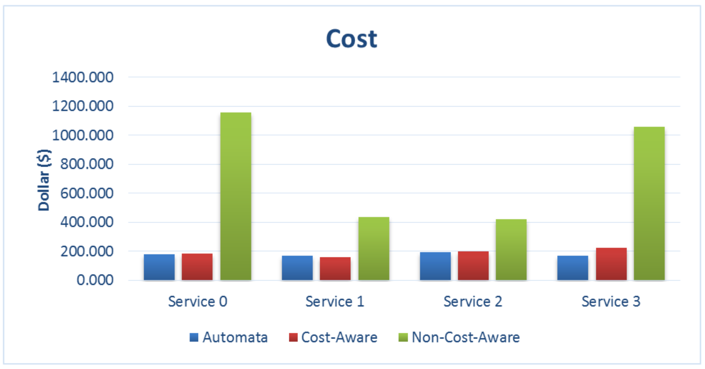

Figure 2. Comparing Cost of Different Approaches

Cost is one important factor for user. The user is trying to run his/her request with the lowest possible cost. In Figure 2, results of simulation have been offered according to the cost of scaling for all three compared approaches for 4 existed services in 24 hours. As you can see random approach has the worst result in scaling 4 mentioned services regarding to its greedy nature of performance as this approach selects its choice on the basis of current needs without regarding cost parameters whenever it needs to perform the scaling. Cost-aware non-automata approach has better results than the random approach in response to all requests. However it is obvious cost-aware approach based on learning automata can decrease costs of scaling substantially. Our proposed approach can be more successful in decreasing scaling expenses in considered services using reward and penalty technique.

In Figure 3, we have compared general cost of applying three approaches. According to the results of the comparison, our proposed approach has been able to improve scaling costs up to 85% compared to random approach and 20 percent compared to cost-aware approach.

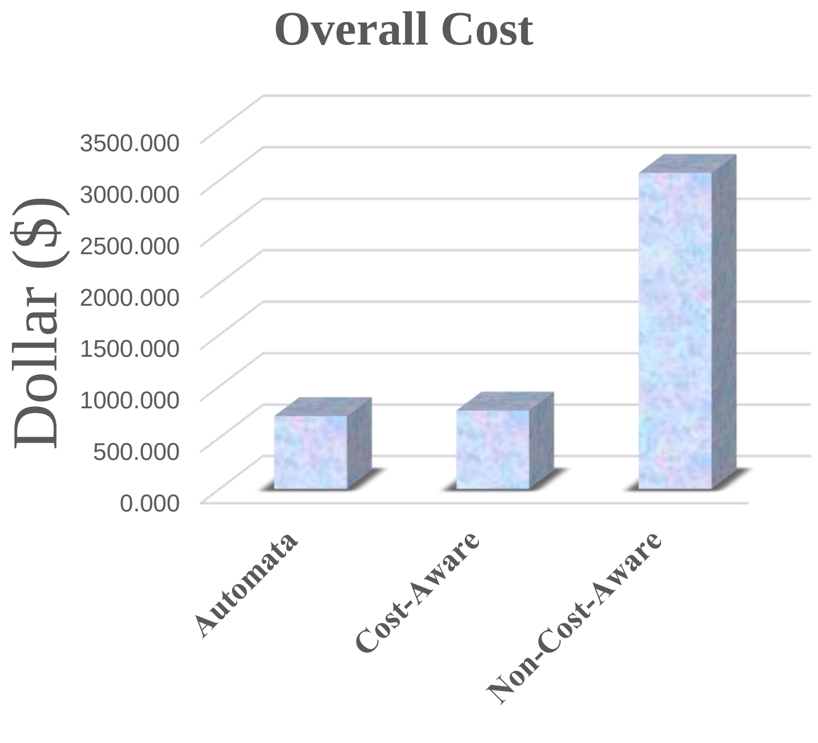

Figure 3. Comparing Overall Cost of Different Approaches

5. Conclusion and Further Work

Cloud services are distributed infrastructure which develops services and communications. Scaling as one of the most important features of cloud computing, tries to allocate and pay back resources based on the requirements. Generally, we expect an efficient scaling mechanism will guarantee the quality of cloud services by the minimum cost. Important factor that have examined in the proposed approach is the cost of scaling. Using optimized virtual resources causes to decrease cost of a cloud service. On the other hand, efficiency improvement will be provided by using appropriate resources fixed by requests which cause to have higher costs. Based on the evaluation results, the proposed scaling approach which has been offered based on learning automata can cost of scaling it to run on demand to optimize the cloud. Focus on scaling our approach is cost optimization. It should be noted that the deadline is applicable only for works in which the performance means efficiency not their catastrophic results. You can also use automatically scaling based on automata for Data intensive applications.

References

[1] R. Anandhi and K. Chitra, "A challenge in improving the consistency of transactions in cloud databases-scalability", International Journal of Computer Applications, vol. 52, no. 2 (**2012**), pp. 12-14.

[2] A. Fereydooni, M. G. Arani and M. Shamsi, "EDLT: An Extended DLT to Enhance Load Balancing in Cloud Computing", International Journal of Computer Applications, vol. 108, no. 7, (**2014**), December, pp. 6-11.

[3] I. Alam, M. Pandey and S. S. Rautaray, "A Comprehensive Survey on Cloud Computing", IJITCS, vol. 7, no. 2, (**2015**), pp. 68-79, DOI: 10.5815/ijitcs.2015.02.09.

[4] L. M. Vaquero, L. Rodero-Merino and R. Buyya, "Dynamically scaling applications in the cloud", ACM SIGCOMM Computer Communication Review, vol. 41, no. 1, (**2011),** pp. 45-52.

[5] M. Mao, J. Li and M. Humphrey, "Cloud auto-scaling with deadline and budget constraints", In Grid Computing (GRID), 2010 11th IEEE/ACM International Conference on, IEEE, (**2010**), pp. 41-48.

[6] M. Ming and M. Humphrey, "Auto-scaling to minimize cost and meet application deadlines in cloud workflows", In Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis, ACM, (**2011**), p. 49.

[7] M. Fallah, M. G. Arani and M. Maeen, "NASLA: Novel Auto Scaling Approach based on Learning Automata for Web Application in Cloud Computing Environment", International Journal of Computer Applications, vol. 113, no. 2, (**2015**), March, pp. 18-23.

[8] M. Fallah and M. G. Arani, "ASTAW: Auto-Scaling Threshold-based Approach for Web Application in Cloud Computing Environment", International Journal of u- and e- Service, Science and Technology (IJUNESST), vol. 8, no. 3, (**2015**), pp. 221-230.

[9] Z. Hasan Masum, E. Magana, A. Clemm, L. Tucker and S. L. D. Gudreddi, "Integrated and autonomic cloud resource scaling", InNetwork Operations and Management Symposium (NOMS), 2012 IEEE‹ pp. 1327-1334.

[10] R. Han, G. Li, M. M. Ghanem and Y. Guo, "Lightweight resource scaling for cloud applications", In Cluster‹ Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM International Symposium on, pp. 644-651.

[11] M. Maurer, I. Brandic and R. Sakellariou, "Enacting SLA's in clouds using rules", In Euro-Par 2011 Parallel Processing, Springer Berlin Heidelberg, (**2011**), pp. 455-466.

[12] D. Xavier, N. Rivierre, A. Moreau, J. Malenfant and I. Truck, "From data center resource allocation to control theory and back", In Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on, pp. 410-417.

[13] K. S. Narendra and M. A. L. Thathachar, "Learning automata: an introduction", Courier Corporation, (**2012**).

[14] D. Menasc and E. Casalicchio, "A Framework for Resource Allocation in Grid Computing", In Proc. of the 12th Annual International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems, (**2004**), pp. 259-267.

[15] N. Chohan, C. Castillo, M. Spreitzer, M. Steinder, A. Tantawi and C. Krintz, "See Spot Run: Using Spot Instances for MapReduce Workflows", In 2nd USENIX Workshop on Hot Topics in Cloud Computing, HotCloud2010. Boston, MA, (**2010**) June.

[16] Y. Yazir, C. Matthews, R. Farahbod and S. Neville, "Dynamic Resource Allocation in Computing Clouds using Distributed Multiple Criteria Decision Analysis", In 3rd International Conference on Cloud Computing, Miami, Florida, USA, (**2010**).

[17] N. Ro, A. Dubey and A. Gokhale, "Efficient autoscaling in the cloud using predictive models for workload forecasting", In Cloud Computing (CLOUD), 2011 IEEE International Conference on, IEEE, (**2011**), pp. 500-507.

[18] S. Abdelwahed, J. Bai, R. Su, and N. Kandasamy, "On the application of predictive control techniques for adaptive performance management of computing systems", Network and Service Management, IEEE Transactions on, vol. 6, no. 4, (**2009**) December, pp. 212 –225.

[19] S. Abdelwahed, N. Kandasamy, and S. Neema, "A controlbased framework for self-managing distributed computing systems", in WOSS '04: Proceedings of the 1st ACM SIGSOFT workshop on Self-managed systems. New York, NY, USA: ACM, (**2004**), pp. 3–7.

[20] T. C. Chieu, A. Mohindra, A. A. Karve, and A. Segal, "Dynamic scaling of web applications in a virtualized cloud computing environment", In e-Business Engineering, 2009, ICEBE'09. IEEE International Conference on, (**2009**), IEEE, pp. 281-286.

[21] J. Yang, C. Liu, Y. Shang, B. Cheng, Z. Mao, C. Liu, L. Niu, and J. Chen, "A cost-aware auto-scaling approach using the workload prediction in service clouds", Information Systems Frontiers, vol. 16, no. 1 (**2014)**, pp. 7-18.

[22] K. Narendra and M. A. L. Thathachar, “Learning Automata: An Introduction”, Prentice Hall, Englewood Cliffs, New Jersey, (**1989**).
[23] K. Najim and A. S. Poznyak, “Learning Automata: Theory and Application”, Tarrytown, NY: Elsevier Science Ltd., (**1994**).
[24] B. S. Taheri, M. G. Arani and M. Maeen, “ACCFLA: Access Control in Cloud Federation using Learning Automata”, International Journal of Computer Applications, vol. 107, no. 6, December (**2014**), pp. 30-40.
[25] 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, (**2011),** pp. 23-50.
[26] J. Yang, C. Liu, Y. Shang, B. Cheng, Z. Mao, C. Liu, L. Niu, and J. Chen, “A cost-aware auto-scaling approach using the workload prediction in service clouds”, Information Systems Frontiers, vol. 16, no. 1 (2**014),** pp. 7-18.

Authors

Khosro Mogouie received the B.S.C degree in Software Engineering from University Azad Arak, Iran in 2009, and M.S.C degree from Azad University of Mahallat, Iran in 2014, respectively. Her research interests include Cloud Computing, Distributed Systems and Cloud Scalability technology.

Mostafa Ghobaei Arani received the B.S.C degree in Software Engineering from IAU Kashan, Iran in 2009, and M.S.C degree from Azad University of Tehran, Iran in 2011, respectively. He’s currently a PhD Candidate in Islamic Azad University, Science and Research Branch, Tehran, Iran. His research interests include Grid Computing, Cloud Computing, Pervasive Computing, Distributed Systems and Software Development.