International Journal of Electrical and Computer Engineering (IJECE)
Vol. 8, No. 6, December 2018, pp. 5359~5370
ISSN: 2088-8708, DOI: 10.11591/ijece.v8i6.pp5359-5370

Prediction Based Efficient Resource Provisioning and Its Impact on QoS Parameters in the Cloud Environment

Lata J Gadhavi, Madhuri D Bhavsar
Institute of Technology, Nirma University, India

Article Info

Article history:

Received Jan 23, 2018
Revised Jun 29, 2018
Accepted Jul 24, 2018

Keyword:

Dynamic resource provisioning
Efficient resource provisioning
Resources deployment
Resources exploitation
Workload prediction

ABSTRACT

The purpose of this paper is to provision the on demand resources to the end users as per their need using prediction method in cloud computing environment. The provisioning of virtualized resources to cloud consumers according to their need is a crucial step in the deployment of applications on the cloud. However, the dynamical management of resources for variable workloads remains a challenging problem for cloud providers. This problem can be solved by using a prediction based adaptive resource provisioning mechanism, which can estimate the upcoming resource demands of applications. The present research introduces a prediction based resource provisioning model for the allocation of resources in advance. The proposed approach facilitates the release of unused resources in the pool with quality of service (QoS), which is defined based on prediction model to perform the allocation of resources in advance. In this work, the model is used to determine the future workload prediction for user requests on web servers, and its impact toward achieving efficient resource provisioning in terms of resource exploitation and QoS. The main contribution of this paper is to develop the prediction model for efficient and dynamic resource provisioning to meet the requirements of end users.

Corresponding Author:

Lata J Gadhavi,
Institute of Technology,
Nirma University, Ahmedabad, Gujarat, India.
Email: 12extphde92@nirmauni.ac.in

1. INTRODUCTION

In the cloud computing environment, cloud users often have time-changing requirements for virtual resources. To provision resources for dynamic and uncertain changes in the workload and to react to these changes accordingly, the cloud provider should manage resources based on the requirement of the users. In the present study, a workload prediction model for dynamic and efficient resource provisioning is proposed to manage the number of user requests and the required resources.

One problem with such a resource provisioning scheme is the occurrence of thrashing, in which, due to frequent variation of the workload (number of job requests), machines can be added and released to meet each requirement while satisfying the QoS metrics. Solving this problem requires an ability to predict the incoming workload on the system and to allocate resources *a priori* by using prediction methods for the required resources. The main contributions in this paper are; (a) The design of a prediction mechanism and of the flow of the prediction model for different periods; and (b) The use of prediction methods to determine the workload based on a historical database.

The rest of the paper is organized as follows. Section II presents a summary of the related work in this domain. Section III provides the domain analysis for the workload pattern. Section IV introduces the prediction mechanism. Section V describes the case study used to validate the proposed approach and number of job request requirement for resources. In section VI, we present the evaluation results for the resource provisioning and allocation. Finally, section VII presents the concluding remarks and future scope.

2. RELATED WORK

Cloud computing characterizes the delivery of computation as a service, in which resources, such as the CPU, software, hardware, information, and devices, are granted to users as services through the Internet. The characteristics of cloud computing, such as auto-scaling, the provisioning of services based on demand, and the utility mechanism, has been largely adopted by different analysts [1]. Cloud computing provides a scalable and flexible platform for providing high-performance computing requirements [2], [3], [4]. However, the pool of resource, flexible service provisioning, and elasticity are not the only potentials of the cloud. The cloud computing system provisions a homogeneous and heterogeneous operating system environment by using virtualization.

In the IT industry, multiple cloud providers deliver QoS to end users according to their demand. However, an issue that emerges from the initial to the deployment stage is the reality that the pattern of access to the application by various end users varies frequently. As a result, there is an unpredictable number of users and variable workload during some periods. In a static and inefficient resource provisioning mechanism, during periods of low demand for resources, there will be an overload of obtainable resources, whereas during periods of high demand, the available resources will be inadequate. This issue leads to poor QoS and high costs. A current research challenge is to develop an adaptive method of resource provisioning in terms of cost and performance.

Cloud computing can address the above challenge by provisioning resources dynamically and effectively to end-user applications based on the prediction of workload and resources according to the user request arrival rate and the service response rate. This means that additional resources can be provided during periods of high demand and released during periods of low demand, without loss of QoS to end users [5] The challenge with such a dynamic resource provisioning strategy is the determination of the proper quantity of resources to be set up and provisioned in a particular period to meet the expected QoS for a variable workload. An ideal solution would require the capability to predict the incoming workload and required resources in advance.

The expected outcome is the determination of the number of virtual machines that should be created, configured, and provided to handle the variable workload. This challenge has been addressed through various ways, such reactive [6], proactive [6], and predictive [7] approaches. Effective resource provisioning is not an easy and uncomplicated task. To meet the above requirements, the following criteria should be considered in developing the algorithm: (a) the computation of the user request rate, (b) the minimization of the request rejection rate, (c) the average response time and the number of requests timed out, (d) the percentage of timed out requests (PTOR) for a number of users, and (e) the accurate computation in advance of the resources required for a variable workload. The current research presents an efficient and effective resource provisioning algorithm that uses a prediction technique to provision and remove resources dynamically. A strategy to improve the resource utilization is proposed and compared with the conventional approach.

3. DOMAIN ANALYSIS

The cloud computing format is mainly described by using three service models: (a) Infrastructure as a Service (IaaS), which explains about the resources offering procedures by cloud providers; (b) Platform as a Service (PaaS), which describes how cloud providers provide an entire cloud environment implemented and deployed in a certain programming language for a specific type of applications; and (c) Software as a Service (SaaS), which refers to applications that can be offered to customers according to their need. The deployed cloud models are mainly classified into four forms: (a) public cloud, which is available for everyone; (b) private cloud, which is hosted solely for one industry; (c) community cloud, which is a cloud environment made accessible only to a certain group of industry or individuals in collaboration; and (d) hybrid cloud, which refers to various clouds that are interconnected with the hosted applications deployed in the cloud environment [8]. In the present work, the term “workload” refers to the number of arrived requests to access the resource of the cloud system.

The application or jobs are need to be switched automatically which are accessed by users. In the cloud computing scenario, the workload category, cloud service models, and deployment models are interconnected with each other, as shown in Figure. 1. The application workload pattern describes diverse user behaviors, which result in the utilization of IT resources in variable forms. The workload can be determined based on the number of user requests, the load calculation on the servers, the network traffic, and the data storage [9]. In the cloud computing environment, the resource provisioning depends on the incoming workload [10]. For the prediction of the workload and resources in this study, the workload patterns are categorized based on the incoming workload, which includes the number of user requests, the amount of work over a specified period, the user request arrival rate, and the time between two consecutive requests (inter-arrival time). According to the above characteristics, the workload is categorized into the five types: static, periodic, unpredictable, continuously changing, and once in a lifetime [11].

3.1. Static Workload

The workload for resources with corresponding or equal usage of a hosted application is called static workload. In this workload type, there is no requirement to increase or decrease the processing power, network bandwidth, data storage, or memory because the workload is constant. A static workload does not require the functionality of the cloud, such as elasticity. Private Websites and wikis are examples of sites with a static workload.

3.2. Periodic Workload

Periodic jobs are very common in our daily routine. Yearly income tax pay, monthly utility bills, traffic during rush hours etc are the examples of periodic tasks or jobs. It is observed that in the same interval, many people using these tasks. During such periodic tasks, it is difficult to provide enough resources for the peak load and to handle the unused resources for the non-peak load. This problem leads to over- or under-provisioning of resources to the hosted application. Periodic tasks occur at the same interval of the day, month, or year; however, they consist of a higher number of requests over a peak period. A periodic workload often requires elasticity and scalability to handle the number of requests during the peak intervals. The computimg workload pattern is depicted on Figure 1.

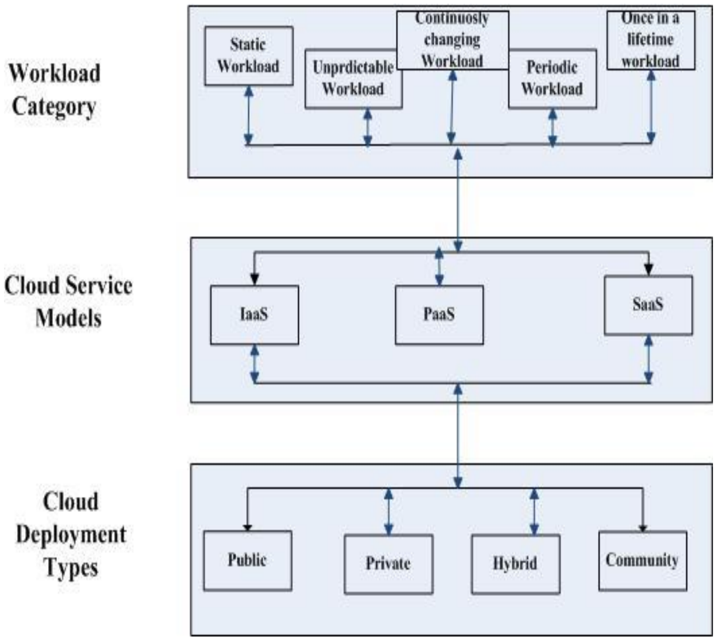

Figure 1. The Cloud Computing Workload Pattern

3.3. Unpredictable Workload

Similarly to a periodic workload, an unpredictable workload consists of increasing and decreasing incoming workloads from the users. This workload is described as unpredictable because the variations occur randomly. As a result, cloud providers have to deal with the unstructured and unplanned provisioning of resources to meet the changing requirements. Accurate prediction is the main challenge to obtaining the scaling requirements for resources for an unpredicted workload. To achieve accurate results, constant observation of the workload is required for example, unpredictable traffic, forecasting etc.

3.4. Continuously Changing Workload

The resource utilization may be maximized or minimized over time, and the workload may change continuously. For many applications, there may be long-term changes in the workload. A continuously changing workload can be considered as a constant escalation or declination of the resource utilization. The elasticity of the cloud enables the provisioning or decommissioning of resources for a continuously changing workload.

3.5. Once-in-a-lifetime Workload

The peak workload utilization may occur only once in a lifetime when the resources are distributed equally. In such a special case, the resource provisioning and decommissioning can often be a manual or dynamic task at a known point in time.

4. PREDICTION MECHANISM

To predict the incoming workload and to identify the workload type, the user requests have to be analyzed. An analysis of the historical database is also needed to meet the resource requirements *a priori*. Based on the user requests and their need analysis are considered for hosted application. Figure 2 shows the procedure of prediction at time t-1 and interval t. The previous period for interval (t-1) and the live period for interval t are observed for a number of active users. The resource requirements for the observed period are monitored, collected, and entered into the historical database after the procedure. Based on the database, the workload for time intervals t-1 and t are calculated, and prediction methods are applied to determine the workload for the next period (t+1).

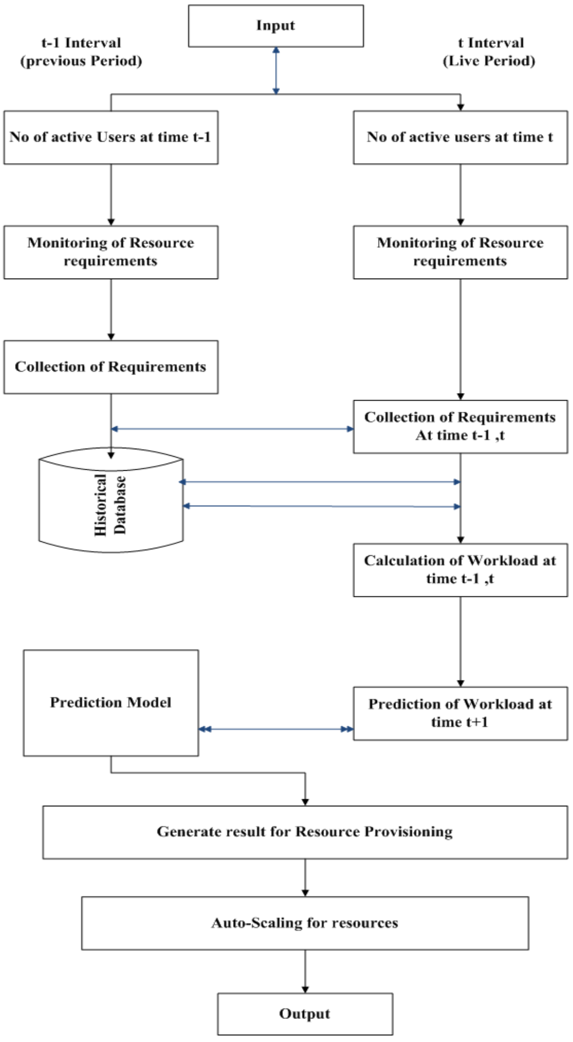
Input
t-1 Interval
(previous Period)
t Interval
(Live Period)
No of active Users at time t-1
No of active users at time t
Monitoring of Resource requirements
Monitoring of Resource requirements
Collection of Requirements
Collection of Requirements At time t-1 ,t
Historical Database
Calculation of Workload at time t-1 ,t
Prediction Model
Prediction of Workload at time t+1
Generate result for Resource Provisioning
Auto-Scaling for resources
Output

Figure 2. The Flow of the Prediction Model

As shown in the above Figure 2, prediction of workload for the resource demands is described in different interval. For time interval $t-1$, the mean CPU, RAM workload, and number of active users are calculated as follows:

$$\begin{aligned}&\alpha^{m}_{t-1(cpu)} = \text{CPU Workload Mean at the interval } t-1\\&\beta^{m}_{t-1(RAM)} = \text{RAM Workload Mean at the interval } t-1\\&\gamma^{m}_{t-1(active\ user)} = \text{Mean number of active users at time } t-1\\&\lambda^{m'}_{t-1} = \text{For whole scenario at time t-1}\end{aligned} \tag{1}$$

For time interval t, the mean CPU, RAM workload, and number of active users are calculated as follows:

$$\begin{aligned}&\alpha^{m}_{t(cpu)} = \text{Mean of CPU Workload at time } t\\&\beta^{m}_{t(RAM)} = \text{Mean of RAM Workload at time } t\\&\gamma^{m}_{t(active\ user)} = \text{Mean of number of active users at time } t\\&\lambda^{m'}_{t} = \text{For whole scenario at time } t\end{aligned} \tag{2}$$

To consider the workload at the initial level, a pseudo-code is described in Algorithm 1.

Algorithm 1: Initial Level

```
Input: Number of requests
Requirement: Calculation of the inter arrival time for the periods $t$-1 and $t$ period
    Begin.
      Monitor active user requests
        While (time interval is $t$-1).
            Calculate the inter arrival time of active users.
              For (each inter arrival time in period $t$-1).
                Monitor workload
                Calculate mean value $\lambda^{m'}_{t-1}$ for week1.
              For (each inter arrival time of $t$ period).
                Monitor the workload.
                Calculate the mean value $\lambda^{m'}_{t}$ for week2.
              End for
              End for
     End.
```

Now for each week of the month, calculate the mean value of the workload and predict for next week based on current and the historical database. Thus, the predicted value for time $t+$ n is:

$$\lambda^{m}_{t+n} = \lambda^{m'}_{t-1} + \lambda^{m'}_{t} + \lambda^{m'}_{t+1} + \cdots + \lambda^{m'}_{t+(n-1)} \tag{3}$$

$$\lambda^{m}_{t+n} = \sum_{t-1}^{t+(n-1)} \lambda^{m'} \tag{4}$$

(3) and (4) express the summation of the previous workload to predict the resource demands for the next period. The pseudo-code for the workload prediction is presented in Algorithm 2.

Algorithm 2: Workload Prediction

```
Input: Number of requests
Requirement: Calculation of the mean CPU, RAM, and number of active users at time interval $t$-1 and
period $t$
  Begin.
    Monitor the number of active user requests at time $t$-1 and period $t$.
    Calculate  $\lambda^{m'}_{t-1}$ and $\lambda^{m'}_{t}$ // by using (1) and (2).
      For each week, calculate the mean workload.
        Predict the workload for the next week $t+$ n,   $\sum_{t-1}^{t+(n-1)} \lambda^{m'}$.
      End for.
   End.
```

To allocate resources to the job, in this section number of job identification are considered for various regions as shown in the Figure. 3.

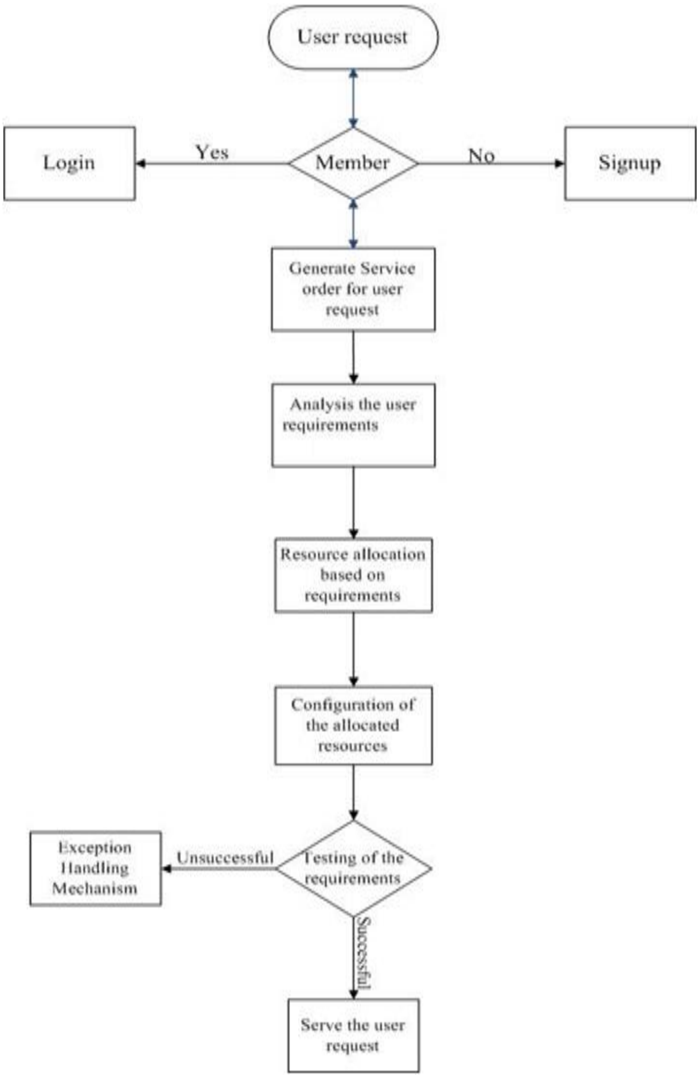
User request
Login
Yes
Member
No
Signup
Generate Service order for user request
Analysis the user requirements
Resource allocation based on requirements
Configuration of the allocated resources
Exception Handling Mechanism
Unsuccessful
Testing of the requirements
Successful
Serve the user request

Figure 3. The Flow of the Resource Allocation

- Determination of the number of requests:

Job profile:

- $J_t = \{j_{i,},\ldots\ldots j_n\}$, where i=1...n, and J_t is number of jobs for the current time
- $j_1 = \{Time(T), Date(D), IPadd(A), Location(L)\}$
- Let $T = (T_1, T_2, T_3, \ldots\ldots.)$ time series
- $D = (d_1, d_2 \ldots.)$ Date of request arrived
- $L = \{(r_i, c_i), \ldots\ldots..(r_n, c_n)\}$, where L is the location of the arrived request
- r, c Region and country, respectively

Total number of job requests from one region= R

$$\sum R = \text{Number of requests per region} \qquad (5)$$

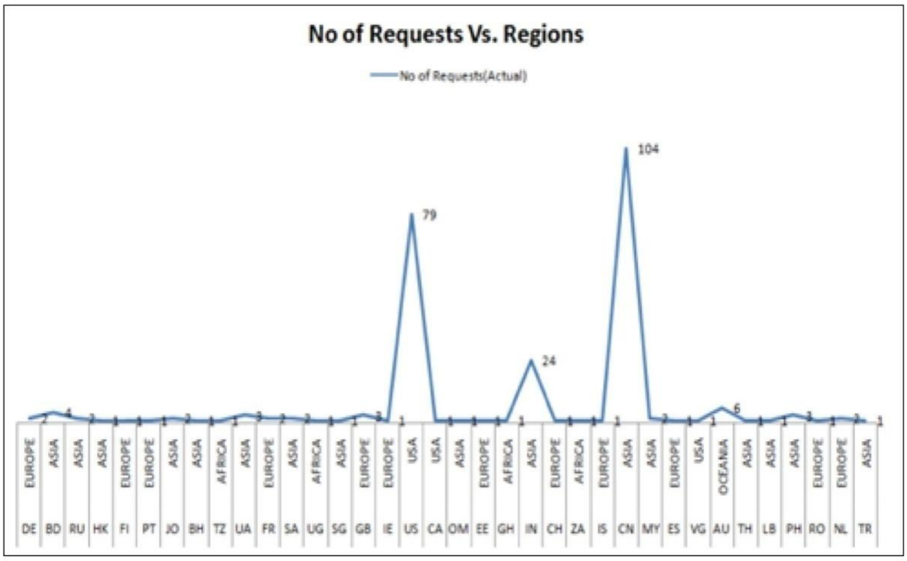

Figure 4. Number of Requests by Region

Figure 4 shows the number of job requests by region (R). Different prediction models are applied for the resource requirements in various periods, including the first and second moving average (FSMA), weighted moving average (WMA), and exponential moving average (EMA) models [11]. The historical data and prediction model output are used as input data, which are trained to improve the accuracy of the resource provisioning strategy. The prediction model uses the output of the predicted value to provision the resources. At the same time, provisioned resources results will be stored in the historical database repository for future prediction. The prediction is done by applying various moving average methods. In particular, the moving average methods FSMA and WMA are recommended to consider non-fluctuations in the short-term demand, as shown in our previous work [12]. An ontology-based dynamic resource provisioning for public cloud was implemented in the previous work [13].

First and Second Moving Average (FSMA) method:

In the first moving average method, the ith user request for resource at time interval t is measured, and N is considered as a moving average period for time interval t [9], as described in (6):

$$F_{t+1} \; = \frac{(y_t\,(i)+y_{t-1}\,(i)+\cdots+\,y_{t-n}\,(i))}{N} \tag{6}$$

Here, y is considered as the original value for each period, and t is regarded as the current period. In (6), F_{t+1} denote the first moving average value for the ith user at time t+1. The resource requirement of the ith user at time t+δ can be predicted as:

$$y_{t+\delta}(\mathrm{i}) = A_t(\mathrm{i}){+}\delta B_t(i) \tag{7}$$

where δ is the predicted time sequence number, and $y_{t+\delta}$ denotes the predicted value at time $y_{t+\delta}$, as shown in (7). Then, to predict the $y_{t+\delta}$ value, the second moving average value can be measured by applying (8):

$$S_{t+1} = \frac{F_{t+1}\,(i)+F_t\,(i)+\cdots+F_{t-(N-1)}\,(i)}{N} \tag{8}$$

In (7), $A_t(i)$ and $B_t(i)$ are calculated by using (9) and (10):

$$A_t(i) = 2F_{t+1}\,(i) - S_{t+1}\,(i) \tag{9}$$

$$B_t(i) = \frac{2F_{t+1}\,(i){-}S_{t+1}\,(i)}{N-1} \tag{10}$$

Based on the above equation, the total number of resource requirements for n users is measured as:

$$y_{t+\delta} = \tag{11}$$

The total number of resources requested by all the n users at time $t + \delta$ can be calculated, and the predicted value at time periods t and $t + \delta$ for for all users can be determined by using equation. (11). Further, the resources are allocated to the users according to their actual need and based on the predicted value, and the results are compared.

Weighted Moving Average Method (WMA):

Whereas the first and second moving average model allocates the same weight to each element of the moving average database, the weighted moving average model allows any weight to be placed on each element, and the sum of all weights equals 1 [11]. Here, the active user requests and the utilization of resources for period F_t are described in (12):

$$F_t = W_i A_{t-i} + W_{i+1} A_{t-(i+1)} + \ldots . W_n A_{t-n} \tag{12}$$

Where F_t = the forecast for the coming period,
n=the total number of periods in the forecast,
$A_{i =}$ the actual occurrence for the period t-i,
Here, $\sum_{i=1}^{n} W_i = 1$

$$WMA \frac{= \sum((Weight\ for\ period\ n)-(demand\ in\ period\ n))}{\sum Weights} \tag{13}$$

Exponential Moving Average Method(EMA):

The exponential moving average (EMA) method is mainly logical and is the easiest approach for predicting fluctuations in the short-term demand. This method is effective for both short-term and time-series prediction. As each new piece of data is added, the oldest observation is dropped, and a new forecast is calculated [14]. The predicted values are calculated by using the smoothing constant α. The EMA method is expressed as:

$$F_t = F_{t-1} + \alpha (A_{t-1} - F_{t-1})$$
$$F_t = \alpha A_{t-1)} + (1 - \alpha) F_{t-1} \tag{14}$$

where F_t is the exponentially smoothed forecast for previous period t
F_{t-1} is the exponentially smoothed forecast made for the prior period,
A_{t-1} = The actual demand in the prior period,
α = The desired response rate or smoothing constant, $0 < \alpha < 1$

This method gives to a higher weight to the later measured value and a lower weight to the earlier measured value. The EMA method is able to respond quickly to fluctuations in the short-term demand. If request from active user is greater than the defined period then it will be considered in the long-term period and if it is less than then it will be considered as a short term period. Long-term and short-term periods are defined as L_t and S_t, respectively.

5. NUMBER OF JOB REQUEST REQUIREMENT FOR RESOURCES

Here, the numbers of requests from various regions are calculated, as shown in (1) and (2). The number of arrived user requests in the deployed cloud for education is evaluated, as shown in our previous work, and the availability of resources in the specified regions is determined. The available resources are generated and monitored by using the commercial Amazon EC2 cloud platform [15, 16, 17]. The capability of the available VMs is calculated; then, the numbers of job requests are allocated to the resources. Once the numbers of jobs are allocated to the VMs, the current load is calculated. If the VM becomes over- or under-loaded, resources are automatically added or released as necessary. The whole procedure is described below.

Let VM={VM_1,VM_2,VM_3,…,VM_N} be a set of N virtual machines and

Task (number of job requests) = {task1, task2, task3, …, K} of K task to be regular and processed in VM.

Here, the fitness of the number of requests (i) may be determined by the capacity of $VM_{j,}$ which is calculated by using (15):

$$fit_{i,j} = \frac{\sum_{i=1}^{n} job\ typelength_{i,j}}{Evaluatecapacityof\ VM_j(capacityj)} \tag{15}$$

job type length shows the how many tasks are submitted on VM_j with the calculation of capacity of specific VM to handle submitted jobs. The capacity of the VM is measured by applying (16):

$$Capacity_j = Num\_proc_j * num\_MIPS_j + BW\_VM_j \tag{16}$$

where Num_proc is the number of processors in the VM, Num_MIPS indicates the number of MIPS, and BW_VM represents bandwidth of the VM.

$$fit_{i,j} = \frac{\sum_{i=1}^{n} job\ typelength_{i,j} + input\ file\ length}{EvaluateCapacityof\ VM_j(capacity_j)} \tag{17}$$

In (17), the input file length is also considered to determine the length of the job before execution. Tasks are assigned to the virtual machine by using the most effective fitness value from the (17).
Load Calculation: When tasks are submitted to the beneath loaded VM, the present work of all offered VM will be measured by victimization of knowledge which is received from the database [18]. To calculate the deviations in load variance on VMs, the following standard deviation (SD) (18) is used:

$$SD = \sqrt{\frac{1}{N}\sum_{j=0}^{n}(X_j - \bar{X})^2} \tag{18}$$

where X_j is the processing time of the VM, as described in (19):

$$X_j = \frac{\sum_{i=1}^{k} job\ type\ length_i}{capacity_i} \tag{19}$$

Then, the mean processing times for all VMs are calculated by applying (20):

$$X = \frac{\sum_{j=1}^{n} X_j}{n} \tag{20}$$

If the SD of the loaded VM is smaller than or equal to the mean, then the system is in a balanced state. On the other hand, if the SD is higher than the mean, then the system is in imbalance, and the auto-scaling mechanism for resource provisioning will be applied. Here, one VM is considered to be capable of serving 50 requests at a time. Thus, when the number of requests >= 50, then scale up new instance and migrate requests on new instance. It is refreshing arrival of request for every 1 minute. Based on the predicted value, the resources are scaled up or down according to the predicted demand. The pseudo-code for scaling up and down is shown in Algorithm 3.

Algorithm 3: Scaling up and down

```
Begin.
  Monitor $\lambda_{t-1}^{m'}$ and $\lambda_t^{m'}$ for the previous and the current time, respectively.
    Predict workload $\lambda_{t+n}^{m}$ at time $t + n$.
  Let R= ri ∪ ri+1 ∪ ri+2 ∪ ri+3..... ∪ rm before scaling, where R is the set of virtual resources.
  Let $\lambda_{t+n}^{m} = \sum_{t-1}^{t+(n-1)} \lambda^{m'} = x$.
  Let $\lambda_{t+(n-1)}^{m'} = y$.
      If $x > y$, then // the workload increases.
        Add r into the R (scale up the virtual resource);
      else if $x < y$ , then // the workload decreases.
        Remove extra r from R (scale down the virtual resource);
    else.
        Refresh the resources.
End.
```

6. RESULTS OF THE QoS METRICS FOR RESOURCE ALLOCATION

Figure 6 shows the efficient resource provisioning with the use of the prediction model achieves maximum CPU utilization compared with the existing (conventional) cloud. Here, the average CPU utilization is obtained for 50 requests on the conventional cloud and 100 requests on the efficient cloud. The results indicate that the efficient cloud uses a minimum number of instances but provides maximum utilization. Figure 7 shows that an efficient response time is obtained for 50 requests on the conventional cloud and 100 requests on the efficient cloud. Figure 8 indicates that efficient resource provisioning with the use of the prediction model achieves the maximum throughput compared with the existing (conventional) cloud. Here, the average throughput is measured for 50 requests on the conventional cloud and 100 requests on the efficient cloud. The results show that the efficient cloud uses a minimum number of instances but provides the maximum throughput.

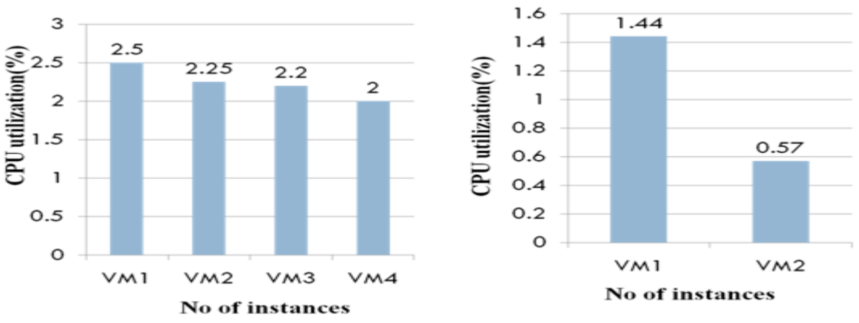

Figure 6. Average CPU utilization for 50 requests on the conventional and 100 requests on the efficient cloud

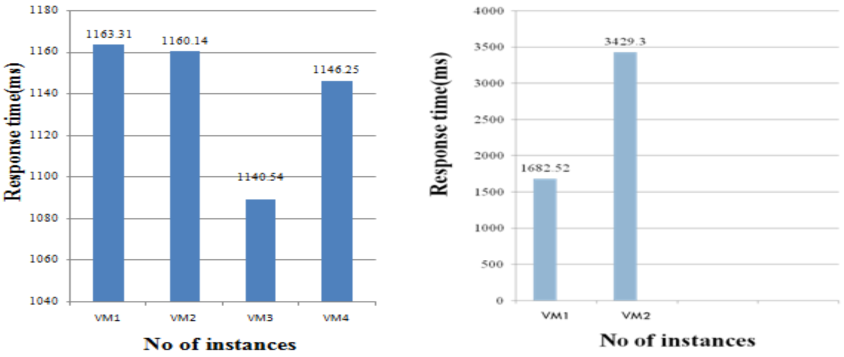

Figure 7. Response time for 50 requests on the conventional cloud and 100 requests on the efficient cloud

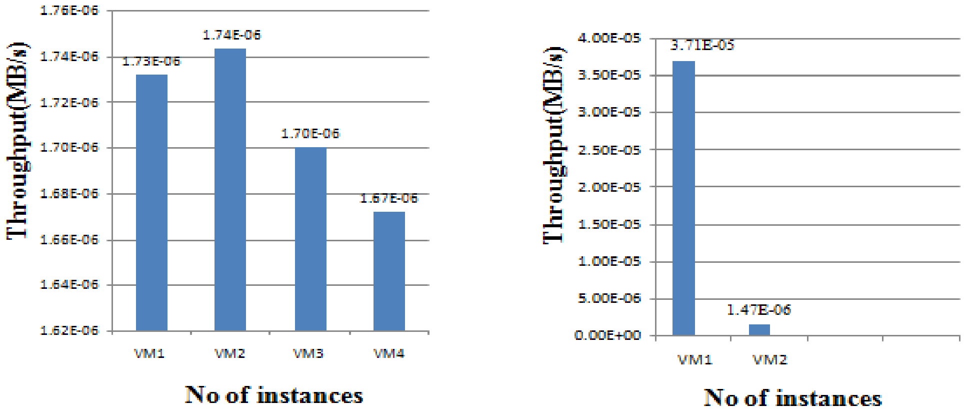

Figure 8. Average throughputs for 50 requests on the conventional and 100 requests on the efficient cloud

7. CONCLUSIONS AND FUTURE WORK

In this research, the design of a prediction model for efficient resource provisioning is presented. A prediction model is an essential part of the cloud environment, in which it is used to predict the number of user requests and the corresponding resource requirements. In the present work, the FSMA, WMA, and EMA models are used to predict the number of user requests for cloud resources. The cloud environment is set up on the EC2 AWS cloud. In addition, the auto-scaling algorithm is used in resource provisioning based on the prediction model. The experimental results of the QoS parameters are presented and compared between the conventional cloud and the efficient cloud before and after prediction. Based on the findings, the proposed prediction model has higher efficiency in resource allocation.

In future works, we will apply the ARIMA model to improve the prediction accuracy for resource provisioning. Further, we plan to integrate the architecture for adaptive resource provisioning by using the workload prediction strategy.

REFERENCES

[1] C. Ji, Y. Li, W. Qiu, U. Awada. “Big data processing in cloud computing environments”, in *Pervasive Systems, Algorithms and Networks’ (ISPAN), 12th International Symposium on.* , pp.17–23. IEEE 2012.

[2] Changlong Li, Hang Zhuang, Kun Lu, Mingming Sun, Jinhong Zhou, Dong Dai, Xuehai Zhou. “An adaptive auto-configuration tool for Hadoop”, *19th International Conference on Engineering of Complex Computer Systems*. Pp.69-72. IEEE. 2014.

[3] Ficco, M. “Security event correlation approach for cloud computing”, *International Journal of High Performance Computing and Networking (IJHPCN)* 7(3). 2013.

[4] M. Banu, A. Aranganathan. “Study of Load Optimization and Performance Issues in Cloud”, *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol 11, no 3, 2018.

[5] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, M. Zaharia. “A view of cloud computing, communication”, vol. 53, no. 4, pp. 50–58. *ACM*. 2010.

[6] Laura R. Moore, Kathryn Bean and Tariq Ellahi. “Transforming Reactive Auto-scaling into Proactive Auto-scaling”, *CloudDP ’13*, ACM. 2013.

[7] Rodrigo N. Calheiros, Enayat Masoumi, Rajiv Ranjan, Rajkumar Buyya. “Workload Prediction Using ARIMA Model and Its Impact on Cloud Applications’ QoS”, *IEEE Transactions on cloud computing*, Vol 3. No.4.IEEE. 2015.

[8] H. Muhasin, R. Atan, M.A. Jabar, S. Abdullah. “The Factors Affecting on Managing Sensitive Data in Cloud Computing”, *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol 11, no 3,2018.

[9] C. Fehling, F. Leymann, R. Retter, W. Schupeck, P. Arbitter. “Cloud Computing Patterns: Fundamentals to Design”, *Build, and Manage Cloud Applications*, vol. XXVI, pp1-367, Springer. 2014.

[10] T. Zaidi, R. Rampratap. “Virtual Machine Allocation Policy in Cloud Computing Environment using CloudSim”, *International Journal of Electrical and Computer Engineering (IJECE)*, vol 8, no 1, 2018.

[11] Petropoulos, F., Kourentzes, N. “Forecast combinations for intermittent demand”, *Journal of the Operational Research Society*, 66 (6), 914–924. 2015.

[12] Lata Gadhavi, Madhuri Bhavsar. “Efficient and Dynamic Resource Provisioning Strategy for Data Processing Using Cloud Computing”, *International Review on Computers and Software (I.RE.CO.S.)*, Vol. 11, No. 8. 2016.

[13] Lata Gadhavi, VIshakha Modi, Madhuri Bhavsar, "Proposed Ontology Framework for Dynamic Resource Provisioning on Public Cloud", *International Journal of Advanced Research in Engineering and Technology (IJARET)* Volume 7, Issue 2, March-April 2016, pp. 118–131.

[14] Fiorucci, J. A., Pellegrini, T. R., Louzada, F., Petropoulos, F., Koehler, A. B. "Models for optimising the theta method and their relationship to state space models", *International Journal of Forecasting*, 32 (4), 1151–1161. 2016.

[15] K. Kalyana Chakravarthi, Vaidhehi Vijayakumar. "Workflow Scheduling Techniques and Algorithms in IaaS Cloud: A Survey", *International Journal of Electrical and Computer Engineering (IJECE)*, vol 8, no 2, 2018.

[16] Marco A. S. Netto, Rodrigo N. Calheiros, Eduardo R. Rodrigues, Renato L. F. Cunha, and Rajkumar Buyya. "HPC Cloud for Scientific and Business Applications: Taxonomy, Vision, and Research Challenges". *ACM Comput. Surv.* 51, 1, Article 8, 2018.

[17] Lindstrom, J. Hermanson, A. Blomstedt, F. Kyosti, P. "A Multi-Usable Cloud Service Platform: A Case Study on Improved Development Pace and Efficiency". *Appl. Sci.*, *8*, 316, 2018.

[18] S. K. Suresh, J.R. Kannan. "Review of Advancements in Multi-tenant Framework in Cloud Computing", *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol 11, no 3, 2018.