DOI 10.12694/scpe.v20i2.1510
Scalable Computing: Practice and Experience ISSN 1895-1767
Volume 20, Number 2, pp. 207–222. http://www.scpe.org © 2019 SCPE

TRIANGULATION RESOURCE PROVISIONING FOR WEB APPLICATIONS IN CLOUD COMPUTING: A PROFIT-AWARE APPROACH

PARMINDER SINGH [*], POOJA GUPTA [†], AND KIRAN JYOTI [‡]

Abstract. The elasticity feature of cloud attracts the application providers to host the application in a cloud environment. The dynamic resource provisioning arranges the resources on-demand according to the application workload. The over-utilization and under-utilization of resources can be prevented with autonomic resource provisioning. In literature, the Service Level Agreement (SLA) based, load-aware, resource-aware and user-behavior aware solutions have been proposed. The solutions are rigid for a particular metric which provides benefit either to end users or to the application providers. In this article, we proposed a Triangulation Resource Provisioning (TRP) technique with a profit-aware surplus VM selection policy. This policy ensures the fair resource utilization in hourly billing cycle while giving the Quality of Service (QoS) to the end-users. The proposed technique used time series workload forecasting, CPU utilization and response time in the analysis phase. The experiment results show that the TRP resource provisioning technique is a profit-aware approach for the application providers and ensure the QoS to the end-users.

Key words: Cloud computing, autonomic computing, resource provisioning, workload prediction, web applications, auto-scaling.

AMS subject classifications. 68M20, 68N19

1. Introduction. Cloud computing paradigm provides the Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS) [34]. The Cloud Service Providers (CSPs) provides the VMs through IaaS with different pricing models such as reserved, on-demand and spot-instances such as Amazon EC2 [2]. Furthermore, the Application Service Providers (ASPs) rent the VMs from CSPs and host the applications. These applications provide SaaS such as web applications for job portal. The end users access web applications through internet and web services [26]. The end users requests are dynamic in nature, and expecting the Quality of Service (QoS). The careful design of resource provisioning techniques can overcome the issue of over-provisioning and under-provisioning called resource oscillation [51].

It is crucial to decide the right amount of resources for the cloud applications. The resource estimation is possible during their execution because it depends upon the incoming workload [41]. The workload fluctuation is quite often in web applications due to irregular access to web services. The network forensics can find initial evidence from web access pattern and further investigation carried for the digital forensics [43]. The over-provisioning state is providing more resources as compared to the required resources, thus lead to high renting cost to ASP. While, in under provisioning state the deployed resources are less as compare to required resources, thus cause the Service Level Agreement (SLA) violation [23]. Therefore, the profit-aware resource provisioning mechanism is required for the less renting cost, while proper utilization of resources in order to meet SLA requirements.

The above-mentioned problems can be solved with dynamic resource provisioning technique. It is an effective approach to provide the resources as per historical data of input workload and current resource utilization. The main goal of ASP is to make maximum profit by minimizing the renting cost and SLA penalty.

In our previous work, the workload prediction model has been developed for the web application workload in cloud computing [46]. The prediction of workload is an important parameter for the estimation of resources. In this paper, we devised a triangulation resource provisioning technique for web applications with the combination of proactive and reactive technique to estimate the resources. The main contribution of this article is as follows:

- The resource provisioning mechanism is designed for ASPs for web applications in the cloud.
- The surplus VM selection algorithm is developed with the profit-aware approach.
- The series of the experiment conducted on real workload weblogs on different metrics.

2. Background. The resource provisioning without human intervention required a self-configuration, self-optimization, self-healing and self-protective mechanism [29, 19]. Therefore, the autonomic resource provisioning

[*]Lovely Professional University, Phagwara, India. (parminder.16479@lpu.co.in).
[†]Lovely Professional University, Phagwara, India. (pooja.19580@lpu.co.in).
[‡]Guru Nanak Dev Engineering College, Ludhiana, India. (kiranjyotibains@yahoo.com).

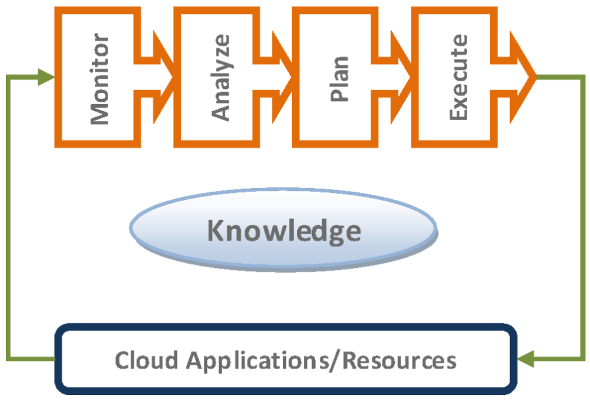

Fig. 2.1. *MAPE-K based cloud resources management.*

is focused on the four phases: Monitoring, Analyzing, Planning and Execution (MAPE) [33] shown in Figure 2.1. The monitoring phase collects the information from the high level and low-level parameters of the cloud environment [17]. Furthermore, the gathered information is analyzed as per reactive, proactive or hybrid approach. Afterward, the planning phase decides the scaling action as scale up/down or scale in/out or either do nothing. Different auto-scaling techniques have been proposed in literature such as threshold rules, fuzzy logic, time-series based, agent-based, etc. [31].

Monitor. The monitor phase fetches the information from the cloud environment about user behavior, SLA violations, and resources utilization [21, 40]. The user decides the threshold, monitoring and scaling intervals, instances bounds, etc. The monitoring performed on the application level, hypervisor level, and VM management.

Analyze. The information collected from the monitoring phase processed for useful inferences to take scaling decision. The reactive and proactive auto-scaler developed by many authors. The reactive approach takes scaling decision on the behalf of current system state. The threshold values are fixed for scale in/out decisions [35]. The predictive approach forecast future CPU utilization or incoming workload, so that the resources could be prepared before the actual workload. The proactive approach could overcome the delayed VM startup problem in the reactive approach.

Plan. The planning phase takes the scaling decisions on the basis of analysis phase parameters. The policies such as resource aware, cost aware, QoS aware, SLA aware and load aware have designed for web application per tier provisioning [27, 11].

Execution. The execution phase took the scaling decision from the planning phase and processed as per the upper and lower resources limit. The VM startup delayed discussed with the client at the time of SLA. The authors [9] designed the executor for web applications in the cloud and, surplus VM selection algorithms have been designed for the scale down decision.

Knowledge. The knowledge is a common repository to be shared with all the four phases of the MAPE loop. It contains the historical logs, policies, and information of resources status [52]. The information is utilized by the autonomic manager.

3. Related Work. In related work section, the brief overview of resource provisioning techniques are discussed.

Huebscher and MacCann [22] published a survey on MAPE knowledge (MAPE-K) and the autonomic computing. The application of autonomic computing introduced with motivation and basic concepts. Singh et al. [47] published a survey and describe the role of cloud in the fog and edge computing. The author mentioned that fog and edge computing can become more beneficial with the carefully designed resource provisioning mechanism in cloud computing. The dynamic cloud-provisioning model [42] proposed for two-tier applications using the MAPE approach. Maurer et al. [32] devised the autonomic control MAPE loop for the adaptive resource provisioning technique for cloud infrastructure. This technique foundation is threshold rule-based, while in our approach time series prediction and reactive rules used for scaling decisions. The control MAPE loop used to design the adaptive framework to configure the scientific applications in an optimized manner [30]. The author [18] designed the resource provisioning approach with using MAPE-K loop.

The linear regression and neural network applied for analysis of resource utilization. The author also discussed the resource provisioning issues in cloud computing [24]. The proactive scaling applied using Double Exponential Smoothing (DES) along with scaling indicator to estimate the utilization of resources [21]. Bankole and Ajila [1, 10] applied a neural network on the analysis phase. The authors analyze the response time, resource utilization and throughput. The web usage mining can applied via machine learning and pattern recognition to find useful inferences [44, 45, 3]. Kaur and Chann [27] designed the analysis and planning phase for resource provisioning. The decision tree has been proposed in order to filter the effective indicators. Herbst et al. [35] devised a MAPE cycle based scaling policies with a reactive approach. The problem of reconfiguration is a challenge in a reactive approach. The author [20] designed the monitoring phase and proactive analysis phase. The prediction method with minimum error selected on the basis of prediction method categorization from simple to complex attributes. The clustering is crucial mechanism for data mining and, without this its becomes difficult to analyze the big data [38]. Singh and Channa [48] performed the workload clustering and allocation of resources for the different QoS feature for the different kind of workload. De Assun et al. [14] devised a scheduling and auto-scaling strategies to find the user patience of end-users in terms of response time to deliver QoS. Toosi et al. [50] devised an auto-scaling algorithm based on threshold-based reactive policy to utilize renewable energy in a load balancing technique.

A rule-based planning phase designed for resources estimation with 5 scaling rules presented for different architectures [13]. The planning phase improved with SLA-aware policy [16]. The proposed model designed to improve the QoS with minimum cost. A planning phase devised using Machine Learning (ML) approach in order to analyze the resource capacity [40]. Fallah et al. [15] applied ML to learn automata in the planning phase with a reactive and proactive auto-scaling approach to plan the resource allocation. The hybrid planning phase designed for auto-scaling with horizontal and vertical scaling techniques. The application cost optimized with autonomic resource provisioning framework [11]. Molto et al. [37] developed the horizontal and vertical scaling hybridization approach with live migration and over-subscription techniques. A framework has been proposed based on the MAPE loop for resource provisioning [5]. The planning phase designed with learning automata technique for resource utilization forecasting. Aslanpour and Dashti [6] proposed the hybrid technique with the combination of reactive and proactive auto-scaling technique. The planning phase has a two-sided policy having one side with SLA violation and the second side is resource utilization. The author [36] analyzed the surplus VMs decisions with cost and cost-aware policies. Author found the response time and resource utilization are two important factors need to consider to release the surplus VMs.

The state-of-the-art articles focused on SaaS resource provisioning. The algorithms, models, mechanism, framework, and architecture have been designed and developed. The resource provisioning technique designed with two methods: Reactive and Proactive. The reactive approach is providing the resources based on threshold-based rules. The major concern in reactive approach is VM startup and setup time, which usually varies from 5 to 15 minutes [27, 4, 8]. This elevates the SLA violation. The proactive method estimates the resources and provides the resources before the incoming workload. Therefore, the proactive technique could decrease the SLA violation.

In this article, the resource provisioning architecture and technique designed for application providers. The proposed mechanism is a combination of reactive and proactive approaches.

4. Proposed Approach.

In this section, the proposed resource provisioning approach is discussed in detail. The devised approach followed the IBMs K-MAPE loop and execute the modules at specific scaling interval. The triangulation approach used a hybrid approach by using proactive and reactive method for resource provisioning. The proposed model is the trade-off between SLA and cost, thus saving the cost while giving the desired QoS. The goal of the proposed technique is to give maximum profit to ASP in term of cost. The cloud architecture for proposed resource provisioning technique is shown in Figure 4.1. The architecture is considered ASP and CSP as part of a cloud environment.

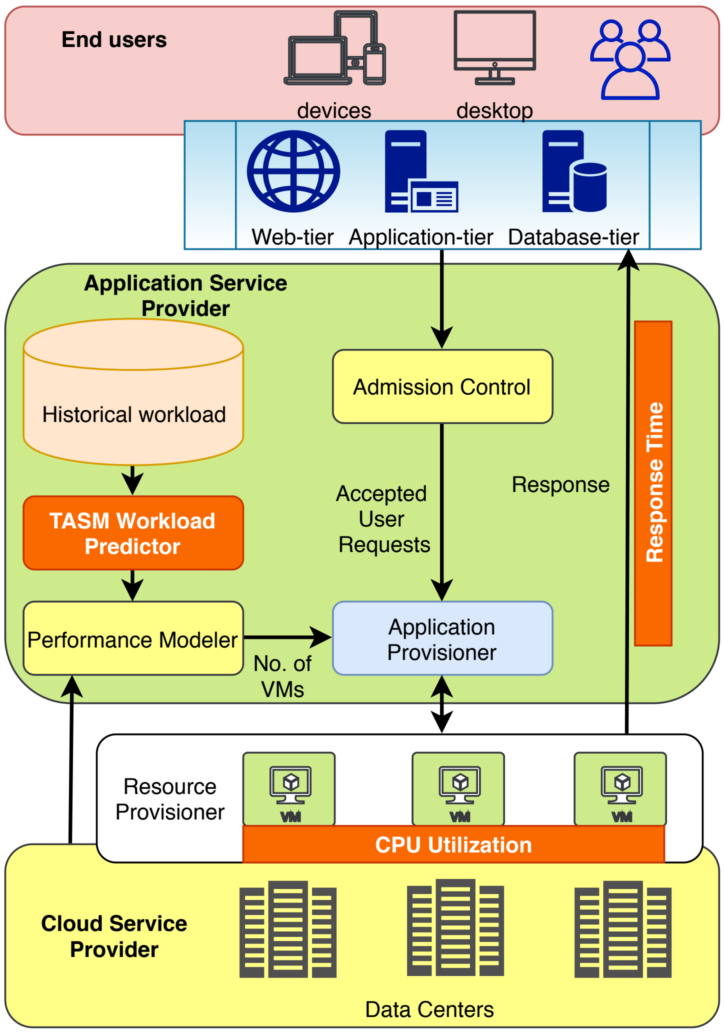

Fig. 4.1. *The cloud resource provisioning architecture.*

4.1. Monitoring Phase. The monitoring phase is generally stored the information about the resource utilization such as CPU utilization called resource-aware approach [51, 29] and, the technique which stores the response time usually called SLA-aware approach. The monitoring of resources, SLA and the incoming requests can take provisioning decision in an effective manner. Figure 4.1 shows the monitoring phase parameters as CPU utilization, response time and incoming workload in each scaling interval.

4.2. Analysis Phase. The analysis phase gives the scaling decision by analyzing the resources, SLA and user behavior, this combination name given as triangulation resource provisioning approach. The performance modeler estimates the number of VMs is working in the analysis phase. The TRP mechanism proposed for effective utilization of resources while giving the response in desired time as mentioned in SLA agreement with the user. Otherwise, the ASP has to pay penalty for the delayed services. The TRP model shown in Figure 4.2. The detailed working of each module of proposed TRP model as per following:

4.2.1. Workload prediction. In our previous work, we developed the time series based Technocrat ARIMA and SVR Model (TASM) [46] for predicting the web application workload. The TASM gives better accuracy as compare to other state-of-the-art prediction models with higher efficiency. In this paper, we used the TASM prediction model to predict the incoming workload. The analytical process for workload forecasting

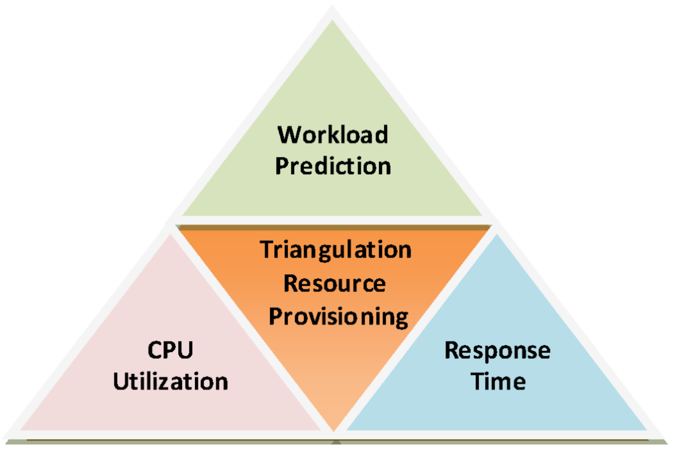

Fig. 4.2. *The proposed triangulation resource provisioning (TRP) approach.*

shown in Figure 4.3. The monitoring phase record the arrival rate and store in the repository named historical workload. Further, the arrival rate is compiled to a discrete time series. The classification on time series performed to select appropriate LR, ARIMA and SVR model. The non-seasonal and seasonal forecasting method applied and fetches the predicted value. The article [46] can be referred for more detail working of the model. In our previous work, short term prediction of 10 minutes is processed. In this work, 1 minute short term prediction has been used to predict the number of requests for a minute.

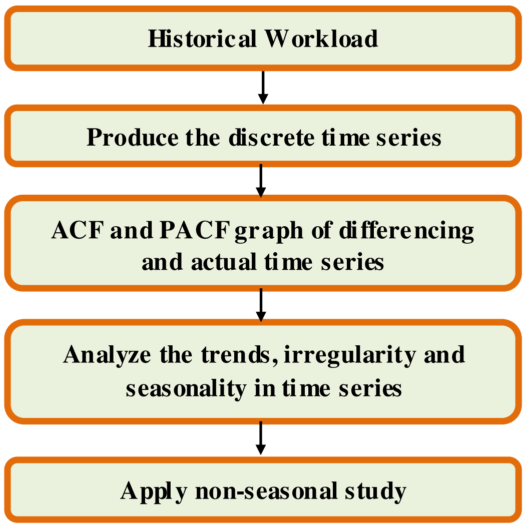
Historical Workload
Produce the discrete time series
ACF and PACF graph of differencing and actual time series
Analyze the trends, irregularity and seasonality in time series
Apply non-seasonal study

Fig. 4.3. *The TASM analytical process for workload forecasting.*

4.2.2. CPU utilization. The resource utilization can be observed from the CPU utilization of each VM deployed for the service of the web application in cloud infrastructure. The average CPU utilization has been calculated as per Eq. 4.1.

$$AverageCpuUtilization = \frac{\sum_{i=1}^{OnlineVms} VM_i CPUutilization}{OnlineVMs} \tag{4.1}$$

where VM_i utilization is the CPU utilization of VM_i and, $OnlineVMs$ are the total VMs including in-service VMs and startup VMs.

$$VM_iCPUutilization = \frac{\sum_{j=1}^{currentcloudlets}(CloudletPEs_j * CloudletLength_j}{PE * MIPS} \tag{4.2}$$

where the number of incoming requests is $CurrentCloudlets$ in the scheduler of VM. The number of processors required is $CloudletPEs$ and number of instruction in each VM in $cloudletLength$. The VM processing elements in number are represented with PE and Million Instruction Per Second ($MIPS$) is the processing capacity of each PE.

4.2.3. Response Time. The response time during the last minute is calculated using Eq. 4.3. It is an important criterion for evaluation of SLA violation. It is further used to select the surplus VM in profit aware approach.

$$AverageResponseTime = \frac{\sum_{i=1}^{TotalProcessedCloudlets} ResponseTime_i}{TotalProcessedCloudlets} \tag{4.3}$$

where $TotalProcessedCloudlets$ are the finished cloudlets in past minute. $ResponseTime_i$ is the delay in i^{th} request's response. The delay is further calculated by using Eq. 4.4.

$$ResponseTime_i = FinishTime_i - ProcessTime_i - ArrivalTime_i \tag{4.4}$$

where the $ArrivalTime_i$ is cloudlet accepted time to the cloud and, $ProcessTime_i$ is the processing time of the cloudlet. $FinishTime_i$ is time recorded from the Clock at finished request.

4.3. Planning Phase. The planning phase applied the rule-based technique to obtain the scaling decision. The compiled values form analysis phase such as predicted workload in scaling interval, average CPU utilization and response time is used in this phase. The threshold values are set for CPU utilization e.g. $20\% - 80\%$ and response time $0.2s - 1.0s$ for lower and upper threshold values respectively.

In the first case, the comparison of the response time and CPU utilization with the defined upper-threshold values, if any observed value greater than upper-threshold values then again forecast values are compared with the current workload. If the current number of *cloudlets* are less as compared to forecast the number of *cloudlets* than $scale - up$.

In the second case, if response time and CPU utilization both are less than lower-threshold values then further evaluate the current cloudlets with forecast cloudlets. If current cloudlets are greater than $scale - down$, otherwise $do - nothing$.

This decision of $scale - up$, $scale - down$ and $do - nothing$ further input to the execution phase.

4.4. Execution Phase. This phase takes the scaling decision scale-up, scale-down or do nothing as an input from the planning phase. The scale-down policy ensure de-provisioning of extra resources in cloud infrastructure. The profit-aware approach for selecting the surplus VM is presented in this paper. The proposed method designed as per the billing cycle of Amazon EC2 instance, which is providing hourly billing cycle. Algorithm 1, show the detailed working of profit-aware surplus VM selection algorithm. The description of notations used in the article is present in Table 4.1.

According to the algorithm 1, the load on the VM and pending minutes in current scaling interval have been considered. The line no. $(4 - 11)$, prepare the $onDemandSurplusVMs$, which are the potential VMs per scaling interval that can scale down. In line no. $(12 - 13)$, if there is only 1 VM select that VM for scale down. In line no. $(14 - 15)$, if there is no VM in the current scaling interval, means every VM having pending service time in current billing hour more than $\triangle s$ than no VM selected as surplus VM. Furthermore, if the number of VMs in $ondemandSurplusVMs$ list is greater than 1, than we have to choose that VM for scale down, whose pending minutes are equals to the processing time required to process the request in the scheduler of that VM. This result in saving the cost as well give the QoS to the end user. In order to do this, line no. $(17 - 29)$ find the VM with least workload. The function $calcuatePendingST$ is calculated the service time of each cloudlet in the scheduler. The line no. $(24 - 27)$ select the VM with least workload. In line no. (30), the algorithm returns the $surplusVM$ to the resource provisioner in CSP.

Table 4.1
Description of notations

Parameter	Description
Clock	Timer for Simulation
$\triangle s$	Scaling Interval
OnDemandVM	Combined List of in-service and pending VMs
VM^P	List of VMs about to ready or in ready state
VM^S	List of VMs in-service
OnDemandSurplusVMs	List of potential surplus VMs in scaling interval
LST	VM's left service time in current billing hour
cloudletList	List of incoming request called cloudlets
SurplusVM	In algorithm 1, the VM finally selected for scale down

Algorithm 1 The pseudo code of proposed profit-aware surplus VM selection policy

```
1: Input: OnDemandVM List ($VM^S + VM^P$)
2: Output: surplusVM
3: Variables: $availableTime, remainingTime$, $onDemandSurplusVMs \leftarrow null$, $minLST \leftarrow anHour$
4: for vm in onDemandVmList do    ▷ Prepare the VMs list having left service time less than or equals to scaling interval
5:     $LST \leftarrow null$;
6:     $availableTime \leftarrow Clock - VM.getRequestTime()$
7:     $LST \leftarrow anHour - (availabletime\%anHour)$
8:     if LST<= $\triangle s$ then
9:         onDemandSurplusVMs.add(vm);
10:    end if
11: end for
12: if onDemandSurplusVMs.legnth() == 1 then    ▷ If list containing only 1 VM
13:     $surplusVM \leftarrow onDemandSurplusVMs.pop()$    ▷ Select the first VM as surplusVM
14: else if onDemandSurplusVMs.empty() then    ▷ If list is empty
15:     surplusVM ← $null$
16: else    ▷ If list containing more than 1 VMs
17:     for vm in onDemandSurplusVMs do
18:         $vm.LST \leftarrow 0$
19:         $cloudletList \leftarrow vm.getScheduler().getCloudletList()$
20:         for cloudlet in cloudletList do
21:             $pendingServiceTime \leftarrow calculatePendingST(cloudlet)$
22:             $vm.LST \leftarrow vm.LST - pendingServiceTime$
23:         end for
24:         if $abs(vm.LST) < minLST$ then    ▷ abs stands for absolute value
25:             $minLST \leftarrow vm.LST$
26:             surplusVM ← vm
27:         end if
28:     end for
29: end if
30: return surplusVm;
```

5. Experiment Evaluation. In this section, the proposed model TRP simulation performed using CloudSim [12], a simulation toolkit for cloud computing infrastructure. R Tool used to implement the prediction model. Furthermore, R Caller library integrates the R Scripts in the CloudSim.

5.1. Experiment Setup. The ClarkNet [49] weblog has been used in the experiment. The various patterns are present in the time series to evaluate the performance of the present algorithm in normal and peak workloads.

Table 5.1
Detail of ClarkNet dataset

Parameter	Value
Log Files	Aug28log, access1
No. of Requests	3,328,587
Duration	2 weeks
Discrete Time series	1 minute

The detail of the workload presented in Table 5.1.

The VMs of bigger size provisioned in case of heavy workload, while small VMs provisioned in case of light workload. The time taken in VM startup considered as 5 minutes. The time shared scheduling policy has been applied in the experiment.

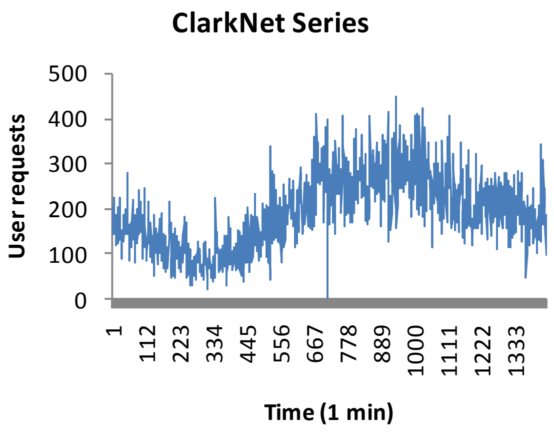

Fig. 5.1. *ClarkNet workload of the* 6th *day in week.*

5.2. Performance Evaluation. The following metrics calculated for performance evaluation:

Response Time: The delay in answering the request is known as response time. The delay in response time is agreed at the time of catering the cloud services. The minimum response time recorded in the cloud environment is 200ms. Some articles [35, 11] have considered the response from 700ms to 1200ms. In our experiment, we considered the 1000ms to ensure the QoS to the end user. The recorded response time utilized for scaling decision in the proposed TRP technique.

SLA Violation: The SLA violation is considered as the delay in response time which is mentioned in the SLA. In our experiments, the desired response time is 1000ms. The percentage of SLA violation calculated to charge the penalty to ASP.

Cost: The resource provisioning mechanism objective is to reduce the cost of application deployment while giving the QoS to the end user. The techniques are able to reduce the cost but ASP has to pay the penalty cost. The sum of rental cost and penalty cost imposed the loss to the ASP. The cost of ASP calculated as the sum of renting cost and SLA penalty.

5.3. Results and Discussion. The performance and efficiency of the proposed TRP mechanism presented in this section.

5.3.1. Workload Prediction. The request arrival rate predicted using the TASM [46] prediction approach. In our previous work, TASM has been proposed and find the prediction accuracy of TASM for the web application is satisfactory as compared to the existing time series prediction model. The TASM is an adaptive prediction model which is the combination of LR, ARIMA and SVR model. These models are selected based on the workload pattern classification. The weblogs first compiled to the discrete time series model, in our previous work the 10 minutes short time prediction has been performed. In this approach, 1-minute short time prediction has been performed and further applied in proposed TRP mechanism. The prediction of ClarkNet series using TASM shown in Figure 5.2.

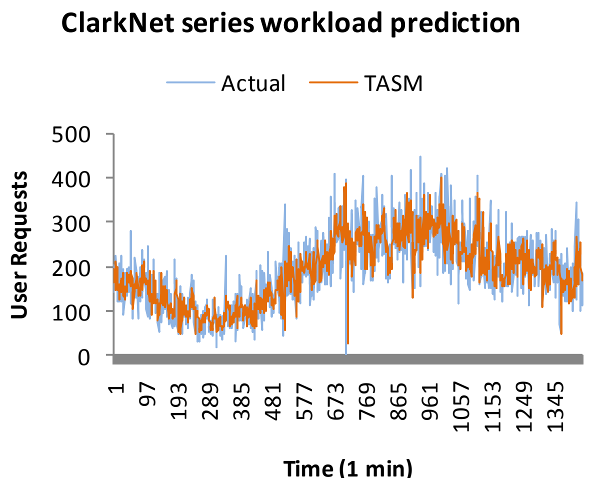

FIG. 5.2. *Workload prediction of ClarkNet series using TASM prediction model.*

$$MAPE = \frac{1}{n}\sum_{s=1}^{n}\left|\frac{actualRequests_s - predictedRequests_s}{actualRequests_s}\right| \times 100\% \tag{5.1}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{s=1}^{n}(actualRequests_s - predictedRequests_s)^2} \tag{5.2}$$

The prediction models accuracy has been calculated with Mean-Absolute-Percentage-Error (MAPE) in Eq. 5.1 and Root-Mean-Square-Error (RMSE) in Eq. 5.2. The experiment results show the RMSE is 42.24 and MAPE as 20.63% of TASM prediction model as compare to LR values are 64.37 and 34.52% and ARMA values are 51.24 and 32.98% receptively for ClarkNet series.

5.3.2. Triangulation Resource Provisioning Mechanism. The TRP mechanism in detail explained in section 4.2. and Figure 4.2. The TRP approach consist of three important parameters to perform the provisioning for web applications in cloud computing. The first approach is related to the resource behavior of the provisioned VMs (singular approach). The monitoring phase records the average CPU utilization after every scaling interval. The threshold based techniques are set upper and lower threshold for provisioning of VM e.g. 80% upper threshold for scale up and 20% lower threshold to scale down decision. The second approach (double approach) add another parameter response time along with CPU utilization. The response time is related to SLA behavior, with which evaluation of SLA violation would be compiled and necessary actions could be taken. Furthermore, the response time is also evaluated on the basis of threshold values. The proposed model is based on the 3-D approach designed by the Aslanpour et al. [8]. The proposed model is known as triangulation resource provisioning approach (TRP) added the third parameter as user-behavior analysis. The future incoming workload is forecasted and considered as the third parameter. The LR, ARIMA and SVR models are used in a TASM prediction model to select the best prediction model based on workload classification. A new profit aware surplus VM selection policy has been proposed to balance the cost to application provider and QoS to the end user. The experiment results shows the performance evaluation of singular-double-TRP mechanisms for resource utilization, QoS and Cost.

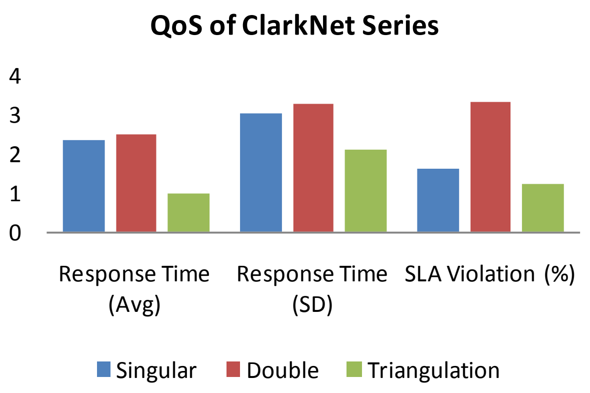

FIG. 5.3. *The QoS of singular, double and proposed triangulation mechanism for ClarkNet Series.*

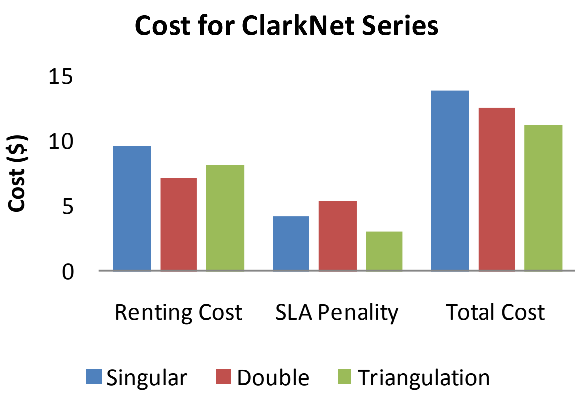

FIG. 5.4. *The cost of singular, double and proposed triangulation mechanism for ClarkNet Series.*

The Figure 5.3 and Figure 5.4 shows the experiment results of Singular, Double and Triangulation resource provisioning. The response time of singular and double technique response time is greater than 1 second which is higher than the required SLA parameter. The proposed triangulation technique is giving QoS as per SLA. The Standard Deviation (SD) of response time depicted by the purposed model assure QoE to the end users thus reduce overall SLA violation percentage. The cost of provisioning is not only the renting cost, but it also combined with the SLA penalty. The purposed triangulation resource provisioning with workload prediction reduces the overall provisioning cost of ClarkNet workload. The proposed TRP model able to reduce the SLA violation and 20% of overall provisioning cost.

5.3.3. Profit-aware surplus VM selection policy. The execution phase in Amazon EC2 randomly selects the VM in the scale-down decision. The policy in state-of-the-art are designed and implement such as Last Up First Down (LUFD) [6], First Up First Down (FUFD) [7], cost-aware and load-aware[9, 8] policies. The super professional executor has been used in the experiment [9]. A new profit-aware surplus VM selection policy presented in this paper. The cost-aware policy selects the VM for scale-down having maximum service time in the current hour regardless the number of requests processed on that VM. If the number of jobs will not complete in the scaling interval than all the jobs will be failed and thus lead to SLA penalty and lead to poor QoE to the end user and, ASP has to pay the penalty cost also. The second technique proposed is load aware policy which selects the VM with the least workload to ensure QoS to the end user. This method can

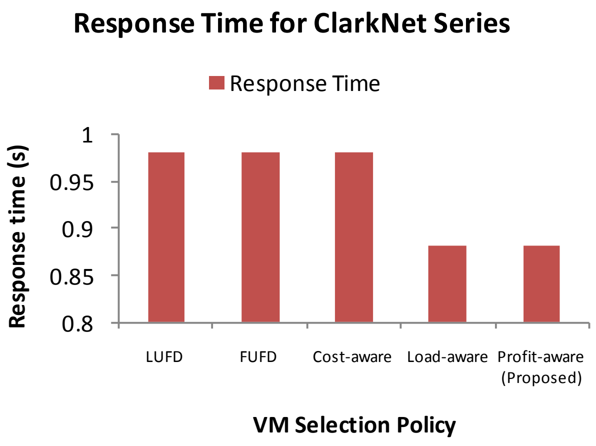

FIG. 5.5. *The response time of surplus VM selection policies for ClarkNet Series.*

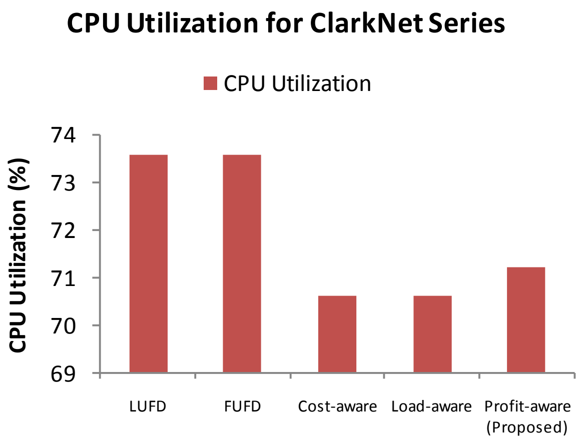

FIG. 5.6. *The CPU utilization of surplus VM selection policies for ClarkNet Series.*

select any machine to de-provision with the least workload may select the VM with maximum minutes pending in current hourly billing cycle thus lead to a more renting cost to the ASP.

In this paper, the profit-aware approach first shortlists the VMs having pending service time less than or equal to the scaling interval. For example, if the scaling interval is 10 minutes, then the VMs having pending service ranges from 0 to 10 minutes has been shortlisted. Afterward, out of the shortlisted VMs, the VM with minimum workload has been selected for the scale down. This process repeated in every scaling interval. The experiment results show that the proposed surplus VM selection policy gives the prominent benefit to the ASP and end user.

We have conducted the experiment with two scenarios on ClarkNet dataset. Both the scenarios are conducted with the proposed TRP mechanism but with the different executors. The first experiment is conducted with ClarkNet workload. The default techniques LUFD and FUFD have been tested with the simple executor. The cost-aware, load-aware and proposed profit-aware surplus VM selection policies are tested with super-professional executor (suprex) [9], which provides the quarantined VM option. The suprex executor will not immediately de-provision the VM selected for scale down. It moves to the quarantined VM list and de-provision

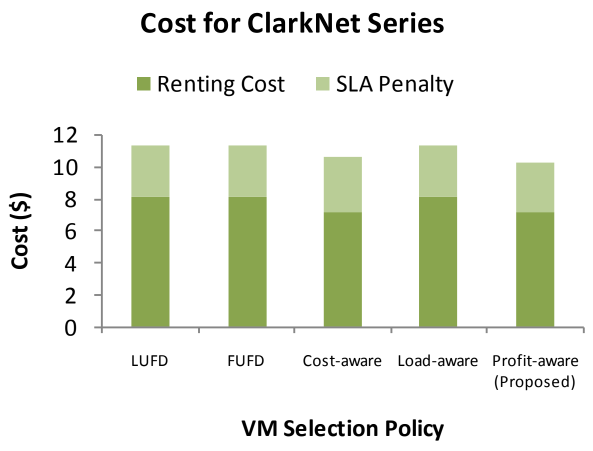

FIG. 5.7. *The cost of surplus VM selection policies for ClarkNet Series.*

when the time period of the hourly billing cycle is complete. In between if VM is required; it can be recalled as in-service VM.

The Figure 5.5, Figure 5.6 and Figure 5.7 shows the comparison of state-of-the-art and proposed profit-aware surplus VM selection policies for response time, CPU utilization and cost metrics respectively. The load-aware and proposed profit-aware policies response time is minimum as compare to other techniques, thus gives better QoE to the end-users with quick response. The LUFD and FUFD techniques gives highest CPU utilization, even the proposed technique gives better CPU utilization from cost-aware and load-aware policies. The LUFD and FUFD is not supported in super-professional executer, which results higher renting cost even due to higher CPU utilization. The proposed VM selection policy gives upto 16% of cost benefit as compare to other surplus VM selection policies. The experiment result shows that the proposed technique gives better QoS with minimum cost.

5.3.4. Comparison of resource provisioning mechanisms. The proposed approach compared with the two baseline approaches named as Cost-aware (LRM) [25] presents prediction based dynamic resource prediction model. The author applied Linear Regression (LR) model for workload forecasting. The second approach under consideration is cost-aware (ARMA) [39]. The second order autoregressivemoving-average (ARMA) model applied for workload forecasting. The third approach is the proposed profit-aware (TRP) resource provisioning model. The TASM prediction model has been used to predict the incoming requests and TRP mechanism is used for resource provisioning and surplus VMs are selected by the profit-aware approach. The strategies under consideration are worked on a proactive resource provisioning model under the MAPE loop.

The CPU utilization of three approaches shown in Figure 5.8 for ClarkNet series at each scaling interval. The simulation result shows the CPU utilization of cost-aware (LRM), cost-aware (ARMA) and proposed profit-aware (TRP). The CPU utilization in some scaling intervals is more than 100%, because the ASP does not have enough resources to process the incoming requests. These conditions lead to SLA violation due to under-provisioning of resources. The result has shown that no technique is able to give 100% CPU utilization. The ARMA and LR have more spikes as compare to profit-aware (ARMA), which clearly indicates that proposed resource provisioning mechanism is successfully overcome the over-provisioning issues for web application in the cloud computing environment.

The delay in request response is shown in Figure 5.9 for three approaches at every scaling interval. The delay in response gives poor QoS to the end users and leads to SLA violation. The ASP has to pay penalty for delay more than decided in SLA agreement. This happens due to under-provisioning and, over-provisioning state having an extra burden of cost to the ASP. The SLA violation minimizes the profit of ASP. In our experiment,

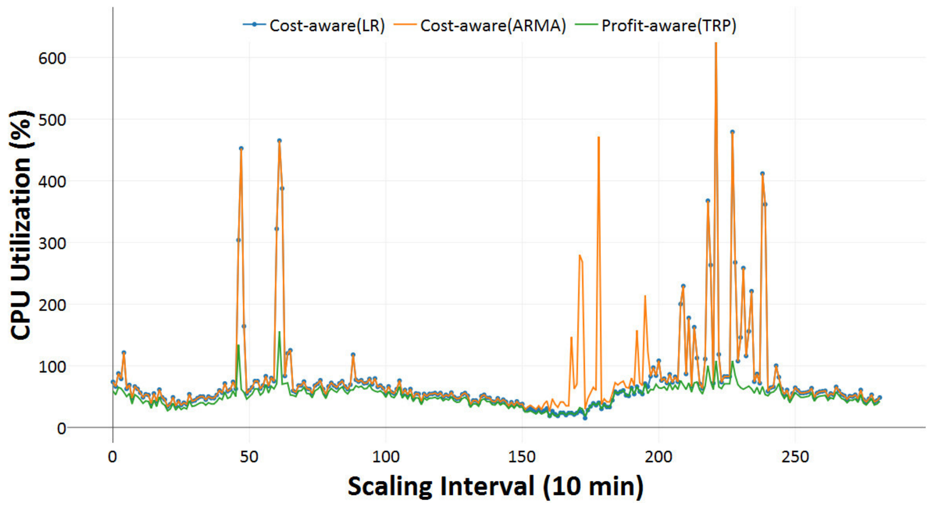

FIG. 5.8. *The comparison of CPU utilization for ClarkNet Series.*

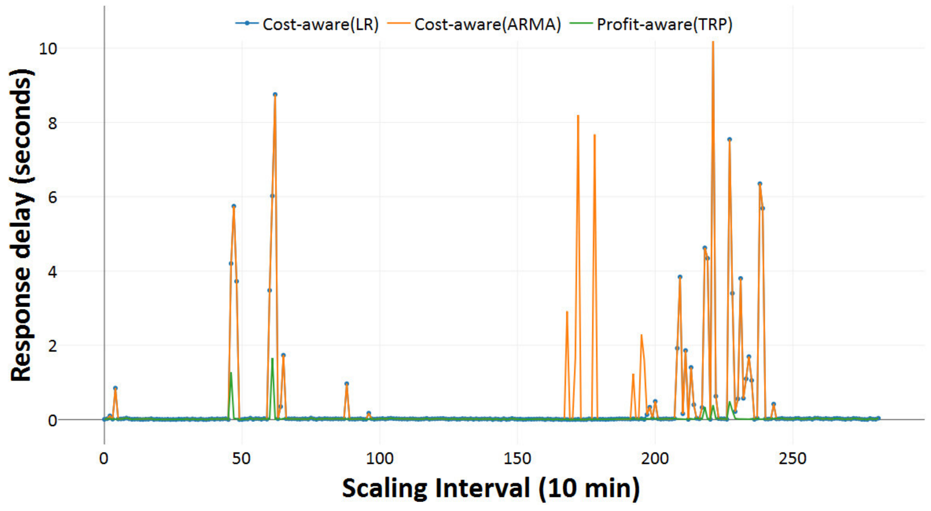

FIG. 5.9. *The comparison of response delay for ClarkNet Series.*

we consider response time as per SLA as $1s$. The Figure 5.9 shows the delay more than $1s$. The comparison result observed the high spikes, which indicates the higher delay in response to the user. The profit-aware approach average delay is equal to 0, only a few small spikes observed. It strongly indicates that the proposed profit-aware (TRP) mechanism is able to give QoS to the end users and reduce the SLA violation.

The VM allocations are shown in Figure 5.10 for three mechanisms for ClarkNet time series at each scaling interval. The cost-aware (LRM) and cost-aware (ARMA) approaches have more under-utilization and over-provisioning states as compare to proposed profit-aware (TRP) approach.

The renting cost and SLA penalty cost of three mechanisms for is shown in Figure 5.11. The renting cost of profit-aware(TRP) approach is higher than cost-aware (LR) and cost-aware(ARMA) model, but when we compared with the SLA penalty, it experienced least SLA penalty. The overall provisioning cost would be less than other provisioning mechanisms. It has been observed that the proposed approach provides the QoS to end-users and comparatively offer 12% more profit to application providers.

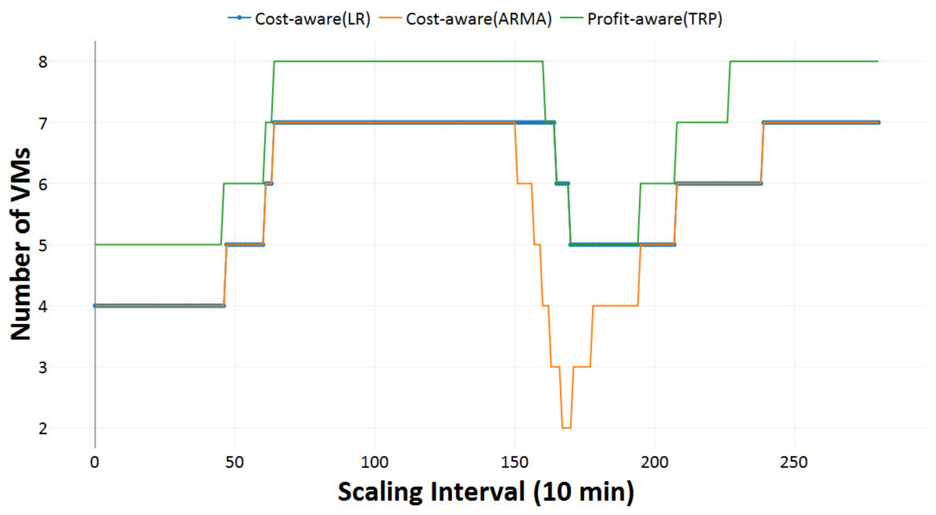

FIG. 5.10. *The comparison of VM allocation for ClarkNet Series.*

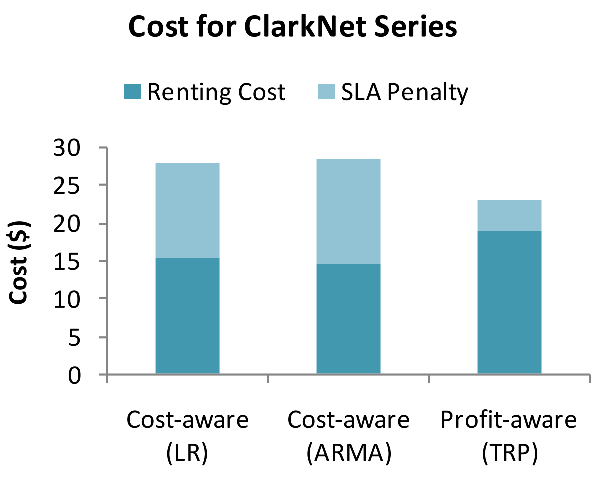

FIG. 5.11. *The comparison of cost for ClarkNet Series.*

6. Conclusion and Future Work. In this article, resource provisioning mechanism presented for web application in cloud computing. The triangulation resource provisioning (TRP) approach has been designed using three parameters: Workload prediction, CPU utilization and response time. We also presented the profit-aware surplus VM selection policy in the scale-down decision in horizontal scaling. The presented model is supporting the MAPE control loop. The workload fluctuations are forecast with the TASM prediction model. The response time, CPU utilization and predicted request are applied in the analysis and planning phase for scaling decisions. The profit-aware surplus VM selection policy used in the execution phase for select the appropriate VM for scale-down. The real workload traces ClarkNet weblog used to evaluate the performance of existing and proposed provisioning mechanisms. CloudSim and R tool used to simulate and obtaining the comparison results. The result shows that the proposed model for web applications provides fair utilization of resources with minimum cost, thus provides maximum profit to application provider and QoE to the end users.

In the future, authors can design new prediction models with higher efficiency. The vertical scaling can be combined with horizontal scaling to mitigate the VM start-up delay issue. The new surplus VM selection policies can be designed for a multi-cloud environment.

REFERENCES

[1] Ajila, S. A. and Bankole, A. A.(2013). Cloud client prediction models using machine learning techniques. In *Computer Software and Applications Conference (COMPSAC), 2013 IEEE 37th Annual*, pages 134–142. IEEE.

[2] Al-Roomi, M., Al-Ebrahim, S., Buqrais, S., and Ahmad, I. (2013). Cloud computing pricing models: a survey. *International Journal of Grid and Distributed Computing*, 6(5):93–106.

[3] Alzubi, J., Nayyar, A. and Kumar, A.(2018). Machine learning from theory to algorithms: an overview. *Journal of Physics: Conference Series*, 1142(1):012012.

[4] Antonescu, A.-F. and Braun, T.(2016). Simulation of sla-based vm-scaling algorithms for cloud-distributed applications. *Future Generation Computer Systems*, 54:260–273.

[5] Arani, M. G., Shamsi, M., et al. (2015). An extended approach for efficient data storage in cloud computing environment. *International Journal of Computer Network and Information Security*, 7(8):30.

[6] Aslanpour, M. S. and Dashti, S. E.(2016). Sla-aware resource allocation for application service providers in the cloud. In *Web Research (ICWR), 2016 Second International Conference on*, pages 31–42. IEEE.

[7] Aslanpour, M. S. and Dashti, S. E.(2017). Proactive auto-scaling algorithm (pasa) for cloud application. *International Journal of Grid and High Performance Computing (IJGHPC)*, 9(3):1–16.

[8] Aslanpour, M. S., Dashti, S. E., Ghobaei-Arani, M., and Rahmanian, A. A.(2018). Resource provisioning for cloud applications: a 3-d, provident and flexible approach. *The Journal of Supercomputing*, 74(12):6470–6501.

[9] Aslanpour, M. S., Ghobaei-Arani, M., and Toosi, A. N.(2017). Auto-scaling web applications in clouds: A cost-aware approach. *Journal of Network and Computer Applications*, 95:26–41.

[10] Bankole, A. A. and Ajila, S. A. (2013). Cloud client prediction models for cloud resource provisioning in a multitier web application environment. In *Service Oriented System Engineering (SOSE), 2013 IEEE 7th International Symposium on*, pages 156–161. IEEE.

[11] Beltrán, M.(2015). Automatic provisioning of multi-tier applications in cloud computing environments. *The Journal of Supercomputing*, 71(6):2221–2250.

[12] Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A., and Buyya, R.(2011). Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and experience*, 41(1):23–50.

[13] Casalicchio, E. and Silvestri, L.(2013). Mechanisms for sla provisioning in cloud-based service providers. *Computer Networks*, 57(3):795–810.

[14] de Assunção, M. D., Cardonha, C. H., Netto, M. A., and Cunha, R. L.(2016). Impact of user patience on auto-scaling resource capacity for cloud services. *Future Generation Computer Systems*, 55:41–50.

[15] Fallah, M., Arani, M. G., and Maeen, M.(2015). Nasla: Novel auto scaling approach based on learning automata for web application in cloud computing environment. *International Journal of Computer Applications*, 113(2).

[16] García, A. G., Espert, I. B., and García, V. H.(2014). Sla-driven dynamic cloud resource management. *Future Generation Computer Systems*, 31:1–11.

[17] Ghanbari, H., Simmons, B., Litoiu, M., and Iszlai, G. (2011). Exploring alternative approaches to implement an elasticity policy. In *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, pages 716–723. IEEE.

[18] Ghobaei-Arani, M., Jabbehdari, S., and Pourmina, M. A.(2018). An autonomic resource provisioning approach for service-based cloud applications: A hybrid approach. *Future Generation Computer Systems*, 78:191–210.

[19] Gill, S. S., Chana, I., Singh, M., and Buyya, R.(2019). Radar: Self-configuring and self-healing in resource management for enhancing quality of cloud services. *Concurrency and Computation: Practice and Experience*, 31(1):e4834.

[20] Herbst, N. R., Huber, N., Kounev, S., and Amrehn, E.(2014). Self-adaptive workload classification and forecasting for proactive resource provisioning. *Concurrency and computation: practice and experience*, 26(12):2053–2078.

[21] Huang, J., Li, C., and Yu, J.(2012). Resource prediction based on double exponential smoothing in cloud computing. In *Consumer Electronics, Communications and Networks (CECNet), 2012 2nd International Conference on*, pages 2056–2060. IEEE.

[22] Huebscher, M. C. and McCann, J. A.(2008). A survey of autonomic computingdegrees, models, and applications. *ACM Computing Surveys (CSUR)*, 40(3):7.

[23] Iqbal, W., Dailey, M. N., Carrera, D., and Janecek, P.(2011). Adaptive resource provisioning for read intensive multi-tier applications in the cloud. *Future Generation Computer Systems*, 27(6):871–879.

[24] Islam, S., Keung, J., Lee, K., and Liu, A.(2012). Empirical prediction models for adaptive resource provisioning in the cloud. *Future Generation Computer Systems*, 28(1):155–162.

[25] J. Yang, C. Liu, Y. Shang, B. Cheng, Z. Mao, C. Liu(2014) A cost-aware auto-scaling approach using the workload prediction in service clouds. In *Information Systems Frontiers*, 16:7–18.

[26] Joshi, N. and Shah, S.(2019). A comprehensive survey of services provided by prevalent cloud computing environments. In *Smart Intelligent Computing and Applications*, pages 413–424. Springer.

[27] Kaur, P. D. and Chana, I.(2014). A resource elasticity framework for qos-aware execution of cloud applications. *Future Generation Computer Systems*, 37:14–25.

[28] Kaneriya, S., Tanwar, S., Nayyar, A., Verma, J.P., Tyagi, S., Kumar, N., Obaidat, M.S. and Rodrigues, J.J.(2018). Data Consumption-Aware Load Forecasting Scheme for Smart Grid Systems. *2018 IEEE Globecom Workshops (GC Wk-*

shps), pages 1–6. IEEE.

[29] Kim, H., el Khamra, Y., Jha, S., and Parashar, M.(2009). An autonomic approach to integrated hpc grid and cloud usage. In *e-Science, 2009. e-Science'09. Fifth IEEE International Conference on*, pages 366–373. IEEE.

[30] Koehler, M.(2014). An adaptive framework for utility-based optimization of scientific applications in the cloud. *Journal of Cloud Computing*, 3(1):4.

[31] Lorido-Botran, T., Miguel-Alonso, J., and Lozano, J. A.(2014). A review of auto-scaling techniques for elastic applications in cloud environments. *Journal of grid computing*, 12(4):559–592.

[32] Maurer, M., Brandic, I., and Sakellariou, R.(2013). Adaptive resource configuration for cloud infrastructure management. *Future Generation Computer Systems*, 29(2):472–487.

[33] Maurer, M., Breskovic, I., Emeakaroha, V. C., and Brandic, I.(2011). Revealing the mape loop for the autonomic management of cloud infrastructures. In *Computers and Communications (ISCC), 2011 IEEE Symposium on*, pages 147–152. IEEE.

[34] Mell, P., Grance, T., et al.(2011). The nist definition of cloud computing.

[35] Mohamed, M., Amziani, M., Belaïd, D., Tata, S., and Melliti, T.(2015). An autonomic approach to manage elasticity of business processes in the cloud. *Future Generation Computer Systems*, 50:49–61.

[36] Moldovan, D., Truong, H.-L., and Dustdar, S.(2016). Cost-aware scalability of applications in public clouds. In *Cloud Engineering (IC2E), 2016 IEEE International Conference on*, pages 79–88. IEEE.

[37] Moltó, G., Caballer, M., and de Alfonso, C.(2016). Automatic memory-based vertical elasticity and oversubscription on cloud platforms. *Future Generation Computer Systems*, 56:1–10.

[38] Nayyar, A. and Puri, V.(2017). Comprehensive Analysis & Performance Comparison of Clustering Algorithms for Big Data. Review of Computer Engineering Research. *Review of Computer Engineering Research*, 4(2):54–80.

[39] N. Roy, A. Dubey, and A. Gokhale(2011). Efficient autoscaling in the cloud using predictive models for workload forecasting. In *International Conference on Cloud Computing (CLOUD)*, pages 500–507. IEEE.

[40] Qavami, H. R., Jamali, S., Akbari, M. K., and Javadi, B.(2013). Dynamic resource provisioning in cloud computing: a heuristic markovian approach. In *International Conference on Cloud Computing*, pages 102–111. Springer.

[41] Qu, C., Calheiros, R. N., and Buyya, R.(2018). Auto-scaling web applications in clouds: A taxonomy and survey. *ACM Computing Surveys (CSUR)*, 51(4):73.

[42] RahimiZadeh, K., AnaLoui, M., Kabiri, P., and Javadi, B.(2015). Performance modeling and analysis of virtualized multi-tier applications under dynamic workloads. *Journal of Network and Computer Applications*, 56:166–187.

[43] Saxena, A.,Shrivastava, G., Sharma, K.(2012). Forensic investigation in cloud computing environment. *International Journal of forensic computer science*, 2:64–74.

[44] Sharma, K., Shrivastava, G. and Kumar, V.(2011). Web mining: Today and tomorrow *2011 3rd International Conference on Electronics Computer Technology*, pages 399–403. IEEE.

[45] Sharma, K. and Bhatnagar, V.(2011). Private and Secure Hyperlink Navigability Assessment in Web Mining Information System *International Journal on Computer Science and Engineering*, 3(6):2245–2250.

[46] Singh, P., Gupta, P., and Jyoti, K.(2018). Tasm: technocrat arima and svr model for workload prediction of web applications in cloud. *Cluster Computing*, pages 1–15.

[47] Singh, S.P., Nayyar, A., Kumar, R. and Sharma, A.(2018). Fog computing: from architecture to edge computing and big data processing. *The Journal of Supercomputing*, pages 1–36.

[48] Singh, S. and Chana, I.(2016). Resource provisioning and scheduling in clouds: Qos perspective. *The Journal of Supercomputing*, 72(3):926–960.

[49] Stephen Balbach ClarkNet web server logs. In *Available: http://ita.ee.lbl.gov/html/contrib/ClarkNet-HTTP.html*, Accessed on: 2019, 20 February.

[50] Toosi, A. N., Qu, C., de Assunção, M. D., and Buyya, R.(2017). Renewable-aware geographical load balancing of web applications for sustainable data centers. *Journal of Network and Computer Applications*, 83:155–168.

[51] Xiao, Z., Song, W., Chen, Q., et al.(2013). Dynamic resource allocation using virtual machines for cloud computing environment. *IEEE Trans. Parallel Distrib. Syst.*, 24(6):1107–1117.

[52] Zareian, S., Veleda, R., Litoiu, M., Shtern, M., Ghanbari, H., and Garg, M.(2015). K-feed-a data-oriented approach to application performance management in cloud. In *2015 IEEE 8th International Conference on Cloud Computing (CLOUD)*, pages 1045–1048. IEEE.

Edited by: Anand Nayyar
Received: Feb 4, 2019
Accepted: Mar 12, 2019

AIMS AND SCOPE

The area of scalable computing has matured and reached a point where new issues and trends require a professional forum. SCPE will provide this avenue by publishing original refereed papers that address the present as well as the future of parallel and distributed computing. The journal will focus on algorithm development, implementation and execution on real-world parallel architectures, and application of parallel and distributed computing to the solution of real-life problems. Of particular interest are:

Expressiveness:
- high level languages,
- object oriented techniques,
- compiler technology for parallel computing,
- implementation techniques and their efficiency.

System engineering:
- programming environments,
- debugging tools,
- software libraries.

Performance:
- performance measurement: metrics, evaluation, visualization,
- performance improvement: resource allocation and scheduling, I/O, network throughput.

Applications:
- database,
- control systems,
- embedded systems,
- fault tolerance,
- industrial and business,
- real-time,
- scientific computing,
- visualization.

Future:
- limitations of current approaches,
- engineering trends and their consequences,
- novel parallel architectures.

Taking into account the extremely rapid pace of changes in the field SCPE is committed to fast turnaround of papers and a short publication time of accepted papers.

INSTRUCTIONS FOR CONTRIBUTORS

Proposals of Special Issues should be submitted to the editor-in-chief.

The language of the journal is English. SCPE publishes three categories of papers: overview papers, research papers and short communications. Electronic submissions are preferred. Overview papers and short communications should be submitted to the editor-in-chief. Research papers should be submitted to the editor whose research interests match the subject of the paper most closely. The list of editors' research interests can be found at the journal WWW site (`http://www.scpe.org`). Each paper appropriate to the journal will be refereed by a minimum of two referees.

There is no a priori limit on the length of overview papers. Research papers should be limited to approximately 20 pages, while short communications should not exceed 5 pages. A 50–100 word abstract should be included.

Upon acceptance the authors will be asked to transfer copyright of the article to the publisher. The authors will be required to prepare the text in $\LaTeX 2_\varepsilon$ using the journal document class file (based on the SIAM's `siamltex.clo` document class, available at the journal WWW site). Figures must be prepared in encapsulated PostScript and appropriately incorporated into the text. The bibliography should be formatted using the SIAM convention. Detailed instructions for the Authors are available on the SCPE WWW site at `http://www.scpe.org`.

Contributions are accepted for review on the understanding that the same work has not been published and that it is not being considered for publication elsewhere. Technical reports can be submitted. Substantially revised versions of papers published in not easily accessible conference proceedings can also be submitted. The editor-in-chief should be notified at the time of submission and the author is responsible for obtaining the necessary copyright releases for all copyrighted material.