Journal of Computer Science 9 (10): 1283-1294, 2013
ISSN: 1549-3636

doi:10.3844/jcssp.2013.1283.1294 Published Online 9 (10) 2013 (http://www.thescipub.com/jcs.toc)

MACHINE LEARNING APPROACHES IN IMPROVING SERVICE LEVEL AGREEMENT-BASED ADMISSION CONTROL FOR A SOFTWARE-AS-A-SERVICE PROVIDER IN CLOUD

[1]R.S. Mohana and [2]P. Thangaraj

[1]Department of Computer Science and Engineering,
Kongu Engineering College, Perundurai, Erode, Tamilnadu, India
[2]Department of Computer Science and Engineering,
Bannari Amman Institute of Technology, Sathyamangalam, Erode, Tamilnadu, India

Received 2013-06-24, Revised 2013-07-15; Accepted 2013-08-22

ABSTRACT

Software as a Service (SaaS) offers reliable access to software applications to the end users over the Internet without direct investment in infrastructure and software. SaaS providers utilize resources of internal data centres or rent resources from a public Infrastructure as a Service (IaaS) provider in order to serve their customers. Internal hosting can ample cost of administration and maintenance whereas hiring from an IaaS provider can impact the service quality due to its variable performance. To surmount these drawbacks, we propose pioneering admission control and scheduling algorithms for SaaS providers to effectively utilize public Cloud resources to maximize profit by minimizing cost and improving customer satisfaction level. There is a drawback in this method is strength of the algorithms by handling errors in dynamic scenario of cloud environment, also there is a need of machine learning method to predict the strategies and produce the according resources. The admission control provided by trust model that is based on SLA uses different strategies to decide upon accepting user requests so that there is minimal performance impact, avoiding SLA penalties that are giving higher profit. Machine learning method aims at building a distributed system for cloud resource monitoring and prediction that includes learning-based methodologies for modelling and optimization of resource prediction models. The learning methods are Artificial Neural Network (ANN) and Support Vector Machine (SVM) are two typical machine learning strategies in the category of regression computation. These two methods can be employed for modelling resource state prediction. In addition, we conduct a widespread evaluation study to analyze which solution matches best in which scenario to maximize SaaS provider's profit. Results obtained through our extensive simulation shows that our proposed algorithms provide significant improvement (up to 40% cost saving) over literature reference ones.

Keywords: SaaS with Machine Learning, Cloud with Machine Learning Techniques, ANN Better Performance than SVM, Machine Learning Associated SLA Based Resource Provisioning

1. INTRODUCTION

Financial Cloud computing is a new paradigm providing usage of applications, platforms, or computing resources such as processing power or bandwidth or storage to customers in a "pay-per-use model". The Cloud model is highly cost-effective as because customers pay only for their actual usage without miscellaneous costs and also scalable as its mainly based changing customer needs. Due to its credits, Cloud has

Corresponding Author: R.S. Mohana, Department of Computer Science and Engineering, Kongu Engineering College, Perundurai, Erode, Tamilnadu, India

been widely adopted in many areas, such as banking, retail industry, e-commerce and academy. Generally, Cloud services can be categorized as: Application (Software as a Service-SaaS), platform (Platform as a Service-PaaS) and hardware resource (Infrastructure as a Service-IaaS).

Here, we focus on the SaaS layer that allows end users to reliably access applications over the Internet without the burden of software related cost and annoying effort (such as software licensing and upgrade). The primary objective of SaaS providers is to minimize cost and maximize Customer Satisfaction Level (CSL). The above mentioned cost includes the administration operation cost, infrastructure cost and finally, penalty cost incurred by SLA violations. CSL depends on degree up to the SLA is satisfied. In general, SaaS providers utilize internal resources of their own datacenters or rent additional resources from a specific other IaaS provider. Internal hosting can create administration and maintenance cost while renting resources from a single IaaS provider can impact the service quality offered to SaaS customers due to the variable performance. To prevail over the above constraints, multiple IaaS providers and admission control are considered over here.

Acquiring resources from multiple IaaS providers lends a large amount of resources with various price schemas and flexible cum varying resource performance to satisfy Service Level Agreement (SLA) specified Service Level Objectives. Here, the admission control has been used as a general mechanism to avoid overloading of resources and SLA satisfaction. But, current SaaS providers do not contain admission control mechanism and their method of scheduling is not known publicly. Thus, the following facts need to be considered to allow efficient use of resources that is offered by multiple IaaS providers, where the resources can be dynamically expanded and reduced on demand:

- Accepting new requests without impacting accepted requests
- Mapping various user requests with different QoS parameters to VMs
- Deciding upon whether the new request should be assigned to available resources or new VM must be initiated

Here, we provide solutions to the above problems by proposing a modern cost-effective scheduling algorithms and admission control technique to maximize the profit of SaaS provider's. These proposed solutions are aimed to maximize the number of the efficient placement of user request on VMs rented from several IaaS providers. We consider various customer's QoS requirements and heterogeneity of infrastructure. The key contributions of this study are fourfold:

- We provided mathematical models for SaaS providers to satisfy customer's requirements
- We proposed admission control and scheduling algorithms for maximizing the SaaS provider's thereby minimizing cost and maximizing CSL
- We utilize the machine learning technique such as SVM and Artificial Neural Network (ANN) to train-up the system for dynamic scenario improving the performance rate of the system
- We evaluate our system to show which methodology of proposed machine learning strategy is best

1.1. Previous Research

Jobs surrendered into a cluster have dynamic requirements depending on user-specific needs and demands. Thus, in utility-driven cluster computing, cluster Resource Management Systems (RMSs) need to be aware of these requirements in order to allocate resources effectively. Service Level Agreements (SLAs) can be used to differentiate different value of jobs as they define service conditions that the cluster RMS agrees to provide for each different job. This Service Level Agreement acts as a bond between a user and the cluster whereby the user is entitled to compensation whenever the cluster RMS fails to deliver the required service. Calheiros *et al*. (2011), they presented a proportional share allocation technique called LibraSLA that takes into account the utility of accepting new jobs into the cluster based on their SLA. They study how LibraSLA performs with respect to several SLA requirements that include: (i) type of deadline to decide whether the job can be delayed, (ii) deadline regarding when the job needs to be finished, (iii) amount to be spent for completing the job and (iv) rate of penalty for compensating the user for failure to meet the deadline. Admission Control is an effective method to avoid overutilization of resources and for meeting user service demands in utility drive computing environment.

Recent emergence of Cloud services and the fame of MapReduce model in Cloud environments make the problem of admission control challenging. Jaideep *et al.* (2010) they proposed a model that allows one to offer MapReduce jobs in on-demand service basis. They presented a learning method based opportunistic

algorithm that accepts MapReduce jobs only when they are unlikely to surpass the overload threshold set by the service provider.

The algorithm proposed in (Jaideep *et al*., 2010) meets deadlines agreed by users in probably more than 80% of cases. They applied an automatically supervised Naive Bayes Classifier for classifying incoming jobs as admissible and non-admissible. From the admissible list of jobs, they then pick a job that is expected to make best use of service provider utility. Also, an external supervision rule automatically evaluates decisions made by the algorithm in review and trains the classifier. They evaluate their algorithm by modeling a MapReduce cluster hosted in the Cloud that offers a set of MapReduce jobs as services to its users. Their results shows that technique of admission control is useful in minimizing failures due to overutilization of resources and by choosing jobs that maximize profit of the service provider.

As cloud computing becomes extensively deployed, progressively cloud services are offered to end users in a way of "pay-as-you-go" manner. Scheduling the dynamic user's service requests in a cost-effective with less SLA violations is one of the most difficult problems of Cloud Service Providers (CSP). To deal with this challenge, in (Lee *et al*., 2010) they first establish a cloud service request model with SLA constraints and then present a new optimization algorithm for profit-driven service request scheduling based on dynamic reuse, which takes account of the personalized SLA characteristics of user requests and current system workload. Their proposed algorithm constructs a on demand resource pool of dynamic virtual machines, attains optimal cloud service request scheduling in sensible time and thus considerably reduces operational costs of cloud service providers thereby increase profits of CSPs. Their simulation experiments show that their proposed algorithm improves virtual resource utilization and increases profits of cloud service providers compared with several baseline algorithms.

A Service Level Agreement (SLA) characterizes an agreement between a particular service resource provider and a user of service. SLAs enclose Quality of Service attributes that must be maintained by a resource provider. These are typically described as a set of Service Level Objectives (SLOs). These attributes need to be measurable and must be monitored during the service provision that has been agreed in accordance to the SLA. The SLA must also contain a set of penalty clauses specifying what happens when service providers fail to deliver the previously agreed quality. Although remarkable work exists on how in the literature, but not much work has focused on actually identifying how SLOs having impact on the specific penalty clauses. The involvement of a trusted mediator is necessary to resolve conflicts between involved parties. The main focus of the study (Reig *et al*., 2010) is on identifying particular penalty clauses that can be associated with an SLA.

Buyya *et al*. (2010), they investigated the method of scheduling user's tasks according to a user-centric value metric called utility or yield. User specified value is smart way for apportioning shared cloud computing resources and it is fundamental to economic approaches for management of resources in linked grids or clusters. Despite that, commonly used batch schedulers do not yet support user specified value-based scheduling and there has been little study of its use in the literature. They introduced heuristics for value-based cloud user task scheduling using a simple formulation of value, in which a task's yield gets decayed linearly with its waiting time. They also have shown the functioning of value-based task scheduling heuristics in a framework for admission control, where clients negotiate for task services. This heuristics balance the risk of future costs against the potential for gains in accepting and scheduling tasks.

In formal method, distributed Support Vector Machines (SVM) algorithms are trained over pre-configured cloud environments to find out an optimally classified solution. These methods are very perplexed and costly for large datasets. Hence, in (Catak and Balaban, 2012) they proposed a method that is mentioned as the Cloud SVM training mechanism (Cloud-SVM) in a cloud computing environment with MapReduce technique for applications of distributed machine learning. Consequently, (i) SVM algorithm is trained in cloud storage servers working concurrently; (ii) Merging all support vectors in each trained cloud node; and (iii) iterate these afore mentioned steps until the SVM converges to the optimal classifier function. Their results of this study are important for training of large scale data sets for machine learning applications. They provided that iterative training of splitted data set in cloud computing environment using SVM will converge to a global optimal classifier infinite iteration size.

Cloud computing facilitates security, privacy and reliable medical data access. Maithili *et al*. (2012) they focuses on cloud computing services that can be extended to medical diagnosis of cancer as well as choice of treatment strategies. An Artificial Neural Network (ANN) judges the possible re-occurrence rate

of tumors correctly in most of the cases by using data obtained from lymphatic node of positive patients. A new framework for cloud computing called User Interface Medical Services (UIMS) is devised. Diagnosis of cancer disease is carried out using ANN and the implementation in cloud environment enhances the efficiency and accuracy of diagnosis.

1.2. System Implementation

1.2.1. System model

Here, we introduce a model of SaaS provider, which consists of "admission control and scheduling system" along with its actors as depicted in **Fig. 1**. The actors are namely users, providers of SaaS and providers of IaaS. The system consists of both application layer and platform layer functions. Cloud users on submitting their QoS requirements request the software from a SaaS provider.

The platform layer uses admission control to infer and analyze the user's requirement of QoS parameters and decide upon whether to accept or reject the request based on the potentiality, availability and price of Virtual Machines (VMs). Then, the scheduling component is responsible for allotting resources that is based on admission control decision. There are two SLA layers with both users and resource providers, that are denoted as SLA (U) and SLA(R) respectively.

1.3. Actors

1.3.1. User

In user's side, a request for application is sent to a SaaS provider's application layer with QoS constraints, such as budget, deadline and penalty rate. Then "admission control and scheduling" algorithms are utilized to admit or reject this request. If the request is accepted, a formal agreement-SLA (U) is signed between both parties such that to guarantee the QoS requirements such as response time that includes the following properties:

- Deadline: Maximal time user would like to wait for the result
- Budget: Amount user is willing to pay for the requested services
- Penalty Rate Ratio: Amount given for consumer's compensation when the SaaS provider misses the deadline
- Input File Size: The size of users input file
- Request Length: Amount of Millions of Instructions (MI) are required to be executed to serve the particular user's request

1.4. SaaS Provider

A SaaS provider hires resources from IaaS providers and leases SaaS to users. SaaS providers aims at minimization of functional cost by efficiently using resources from IaaS providers and improving Customer Satisfaction Level by providing parameters of SLAs, that are used to guarantee QoS requirements of accepted users. From SaaS provider's point of view, there are two layers of SLA with both users and resource providers. It is essential to establish two layers of SLA, because SLA with user can help the SaaS provider to improve the CSL by gaining users trust of the quality of service; SLA with resource providers can enforce resource providers to deliver the satisfied service. When any user in the contract violates its terms, the defaulter has to pay for the penalty according to the clauses defined in the SLA.

1.5. IaaS Provider

An IaaS provider Resource Provider (RP), offers Virtual Machines to SaaS providers and it is in charge for dispatching images of VM to run on their physical resources. The SaaS provider platform layer uses images of VM to create instances. It is necessary to establish SLA with a resource provider called SLA(R), because it enforces the resource provider to guarantee QoS. In addition, it provides jeopardy of transfer for SaaS providers, when the SLA terms are violated by resource provider. The SLA(R) includes the following properties:

- Service Initiation Time: Time taken to deploy a VM
- Price: Amount the SaaS provider has to pay per hour for using a VM from a resource provider
- Input Data Transfer Price: Amount the SaaS provider has to pay for data transfer from local machine (their own machine) to resource provider's VM
- Output Data Transfer Price: Amount the SaaS provider has to pay for data transfer from resource provider's VM to local machine
- Processing Speed: Fast at which VM is processing. Machine Instruction Per Second (MIPS) is used as a unit of a VM's processing speed
- Data Transfer Speed: The fast at which the data is transferred. It relies on the location distance and also the network performance

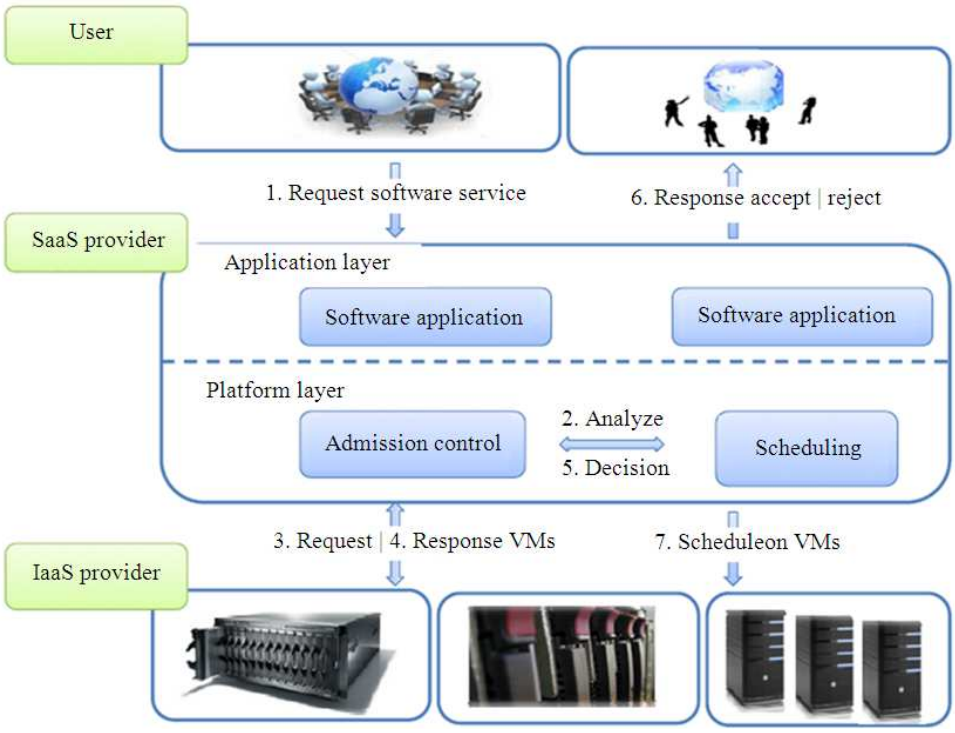

Fig. 1. A high level system model for application service scalability using multiple IaaS providers in Cloud

1.6. Profit model

At a given time instant t, I be the number of initially initiated VMs and total number of IaaS providers is denoted by J. Let IaaS provider j provide N_j types of VM, where each VM type l has P_{jl} price. The prices/GB charged for data transfer-in and -out by the IaaS provider j are $inPri_j$ and $outPri_j$ respectively. Let ($iniT_{ijl}$) be the time taken for initiating VM i of type l. Let us assume a new user submit a service request at submission time $subT^{new}$ to the SaaS provider. This new user offers a maximum price B^{new} (Budget) to SaaS provider with deadline DL^{new} and Penalty Rate β^{new}. Let $inDS^{new}$ and $outDS^{new}$ be the data-in and -out required to process the user requests.

Let $Cost_{ijl}^{new}$ be the total cost incurred to the SaaS provider by processing the user request on VM i of type l and resource provider j. Then, the profit $Prof_{ij}^{new}$ gained by the SaaS provider is defined as Equation 1:

$$Prof_{ij}^{new} = B^{new} - Cost_{ijl}^{new}; \forall i \in I, j \in J, l \in N_j \quad (1)$$

The total cost incurred to SaaS provider for accepting the new request consists of request's processing cost (PC_{ijl}^{new}), data transfer cost (DTC_{ijl}^{new}), VM initiation cost (IC_{ijl}^{new}) and penalty delay cost (PDC_{ijl}^{new}) (to compensate for miss deadline). Hence, the total cost is given by processing the request on VM i of type l on IaaS provider j Equation 2:

$$Cost_{ijl}^{new} = PC_{ijl}^{new} + DTC_{ijl}^{new} + IC_{ijl}^{new} + PDC_{ijl}^{new}; \forall i \in I, j \in J, l \in N_j \quad (2)$$

The processing cost (PC_{ijl}^{new}) for serving the request is dependent on the new request's processing time ($procT_{ijl}^{new}$) and hourly price of VM_{il} (type l) offered by IaaS provider j. Thus, PC_{ijl}^{new} is given by Equation 3:

$$PC_{ijl}^{new} = procT_{ijl}^{new} \times P_{jl}; \forall i \in I, j \in J, l \in N_j \quad (3)$$

Data transfer cost as described below includes cost for both data-in and data-out Equation 4:

$$DTC_{ijl}^{new} = inDS^{new} \times inPri_{jl} + outDS^{new} \times outPri_{jl}; \forall j \in J, l \in N_j \quad (4)$$

The initiation cost (IC_{ijl}^{new}) of VM i (type l) is dependent on the type of VM initiated in the data center of IaaS provider j Equation (5):

$$IC_{ijl}^{new} = iniT_{ijl} \times P_{jl}; \forall i \in I, j \in J, l \in N_j \quad (5)$$

In Equation (6) penalty delay cost (PDC_{ijl}^{new}) is how much the service provider has to give discount to users for SLA (U) violation. It is dependent on the penalty rate (β^{new}) and penalty delay time (PDT_{ij}^{new}) period:

$$PDC_{ijl}^{new} = \beta^{new} \times PDT_{ijl}^{new}; \forall i \in I, j \in J, l \in N_j \quad (6)$$

To process any new request, SaaS provider either can allocate a new VM or schedule the request on an already initiated VM. When the service provider schedules the new request on an already initiated VM_i, the new request has to wait until VM i becomes available. Thus the time for which the new request has to wait until it start processing on VM i is $\sum_{k=1}^{k} procT_{ijl}^{k}$, where K is the number of request yet to be processed before the new request. Thus, PDT_{ij}^{new} is given by Equation (7):

$$\begin{aligned} PDT_{ijl}^{new} &= \{t + \sum_{k}^{k} procTijl^{k} + procTijl^{new} \\ &-DL^{new}, iprocTijl^{new} + iniT_{ijl} + DTTijl^{new} - DL^{new} \end{aligned} \quad (7)$$

DTT_{ijl}^{new} is the data transfer time which is the summation of time taken to upload the input ($inDT_{ill}^{new}$) and download the output data ($outDT_{ijl}^{new}$) from the VM_{il} on IaaS provider j. Thus the time of data transfer is given by Equation (8):

$$DTT_{ijl}^{new} = inDT_{ill}^{new} + outDT_{ijl}^{new}; \forall i \in I, j \in J, l \in N_j \quad (8)$$

Thus, the response time (T_{ijl}^{new}) for the new request to be processed on VM_{il} of IaaS provider j is calculated in Eq. (9) and consists of VM initiation time ($iniDT_{ijl}^{new}$), request's service processing time ($procT_{ijl}^{new}$), data transfer time (DTT_{ijl}^{new}) and penalty delay time (PDT_{ijl}^{new}) Equation (9):

$$\begin{aligned} T_{ijl}^{new} &\sum_{k=1}^{K} procT_{ijl}^{k} procT_{ijl}^{new}, \text{if new VM is not initiate} \\ &= procT_{ijl}^{new} + iniT_{ijl} + DTT_{ijl}^{new}, \text{if new VM is initiate} \end{aligned} \quad (9)$$

The investment return (ret_{ijl}^{new}) to accept new user request per hour on a particular VM_{il} in IaaS provider j is calculated based on the profit ($prof_{ijl}^{new}$) and time (T_{ijl}^{new}) Equation (10):

$$ret_{ijl}^{new} = prof_{ijl}^{new} / T_{ijl}^{new}; \forall i \in I, j \in J, l \in N_j \quad (10)$$

1.7. Support Vector Machines (SVM)

In the field of machine learning, Support Vector Machines (SVM) offers most robust and accurate classification method due to their generalized properties. Having sound theoretical foundation and also proven effectiveness, SVM has successfully applied in many fields. We apply such effective technique to train up our system to choose a best scenario that already happened successfully during training is now followed in testing. This machine learning techniques are more adaptable to our dynamic cloud environments, thereby increasing the percentage of efficiency.

SVM implements the Structural Risk Minimization Principle which seeks to minimize an upper bound of the generalization error, which eventually results in better generalization of SVM than that of traditional techniques. The training of SVM is equivalent to solving a linearly constrained convex quadratic programming problem and therefore the solution of SVM is always globally optimal and free from local minima. In fact, the solution is only determined by the support vectors which are a subset of the training data so that the solution is often very sparse. Another advantage of SVM is that its solution does not depend on a data dimensionality, unlike that of many other methods and this makes it an attractive choice for dealing with high dimensional datasets.

Given that the data cannot be always linearly separated in an input space, SVM performs their mapping into another, a higher dimensional feature space where the data are supposed to be linearly separated. So called "kernel trick" allows not to calculate this mapping explicitly. Instead, the mapping into the feature space is implicitly defined by a kernel function computing the inner product of two feature vectors corresponding to two inputs. In some cases, if the data are noisy, there can be nonlinear separation in the feature space. To deal with this obstacle, the following (dual) optimization problem is to be solved.

Support vector machine is a supervised learning method in statistics and computer science, to analyze data and recognize patterns, used for regression analysis and classification. The standard SVM takes a set of input

data and predicts. In this standard SVM for each of the given input, that is of two possible classes which forms the input that makes SVM a non-probabilistic binary linear classifier. Note that if the training data are linearly separable, we can select the two hyper-planes of the margin in a way that there are no points between them and then try to maximize their distance. With the help of geometry, we used to find the distance between these two hyper-planes is 2/||w||. Given some training data D, a set of n points of the form Equation (11):

$$D = \{(X_i, y_i) \mid X_i \in R^m, y_i \in \{-1,1\}\}_{i=1}^{n} \quad (11)$$

where, X_i is an m-dimensional real vector, y_i is either-1 or 1 denoting the class to which point X_i belongs. SVMs aim to search a hyper-plane that maximizes the margin between the two classes of data in D with the smallest training error. This problem can be formulated as the following quadratic optimization problem Equation (12):

$$\begin{aligned} &\text{minimize}: P(w,b,\xi) = \frac{1}{2}\| w \|^2 \, C\sum_{i=1}^{m}\xi_i \\ &\text{subject to}: y_i((w,\phi(x_i)) + b) \geq 1 - \xi_i \text{ where } \xi_i \geq 0 \end{aligned} \quad (12)$$

For i = 1,…,m, where ξ_i slack variables and the cost of each slack are is denoted by the constant C that is a trade-off parameter which controls minimizing the training error and the maximization of the margin. The decision function of SVMs is $f(x) = w^T \phi(x)+b$ where the w and b are obtained by solving the optimization problem P in (12). By using Lagrange multipliers, the optimization problem P in (12) can be expressed as Equation (13):

$$\begin{aligned} &\text{minimize}: F(\alpha) = \frac{1}{2}\alpha^T Q\alpha^T - \alpha^T 1 \\ &\text{subject to}: 0 \leq \alpha \leq C \text{ where } y^T\alpha = 0 \end{aligned} \quad (13)$$

where, $[Q]_{ij} = y_i y_j \phi^T(x_i)\phi(x_j)$ is the Lagrangian multiplier variable. There is no need of knowing ϕ, but it is necessary to know is how to compute the modified inner product which will be called as kernel function represented as $K(x_i,x_j) = \phi^T(X_i)\phi(X)$. Thus, $[Q]_{ij} = y_i y_j K(x_i,x_j)$. Choosing a positive definite kernel K, then optimization problem P is a convex Quadratic Programming (QP) problem with linear constraints and can be solved in polynomial time.

The SVM predicted on training values are applied to the new user request during testing. This decision chosen according to test data aims in maximizing return of investment and resource provider efficiency minimizing the failures and penalties as represented as **Fig. 2**.

1.8. Working of SVM

In machine learning, Support Vector Machines (SVM) is a supervised learning model with associated learning algorithms that analyze data and recognize patterns which are used for classification. Here we use the SVM to classify the success of ROI while allocating a resource for a user cloud service request. The input to the SVM training encompasses the set of parameters from both user request (Deadline, Budget, Penalty Rate Ratio, Input File Size and Request Length) and the other related information of IaaS provider (Service Initiation Time, Price, Input Data Transfer Price, Output Data Transfer Price, Processing Speed, Data Transfer Speed).

This mapping operation of User request to an available resource or the IaaS provider resource is needed to be analyzed for SVM training using the existing SLA based technique. The input comprises of aforementioned details represented as matrix and denoted by x_i and w is the weight value matrix whose product is summed with bias value to give the class value. This is given by:

$$x_i.w + b = 0$$

This above equation marks a central classifier margin. This can be bounded by soft margin at one side using the following equation:

$$x_i.w + b = 1$$

The input of SVM is always plotted as data points in the graph. Initially during training the weight value is adjusted such that to get the expected outcome i.e., Profit/ROI denoted as binary value true with “1” as per the equation $x_i.w+b = 1$ and “0” denotes the Loss in ROI. This weight value of successful ROI is utilized for testing phase. During testing, the new request x_i+1 is need to be analyzed with previously obtained w with bias value b. If it results in 1 then allocation procedure followed during testing will lead to profit else it may incur a loss. Thus the classified output is given by:

$$\begin{aligned} y_{i+1} &= x_{i+1,}w + b = 1, \text{Profit} \\ & x_{i+1,}w + b = 0, \text{Loss} \end{aligned}$$

This is for when the minimum error is zero and may vary according to initial setting of parameters.

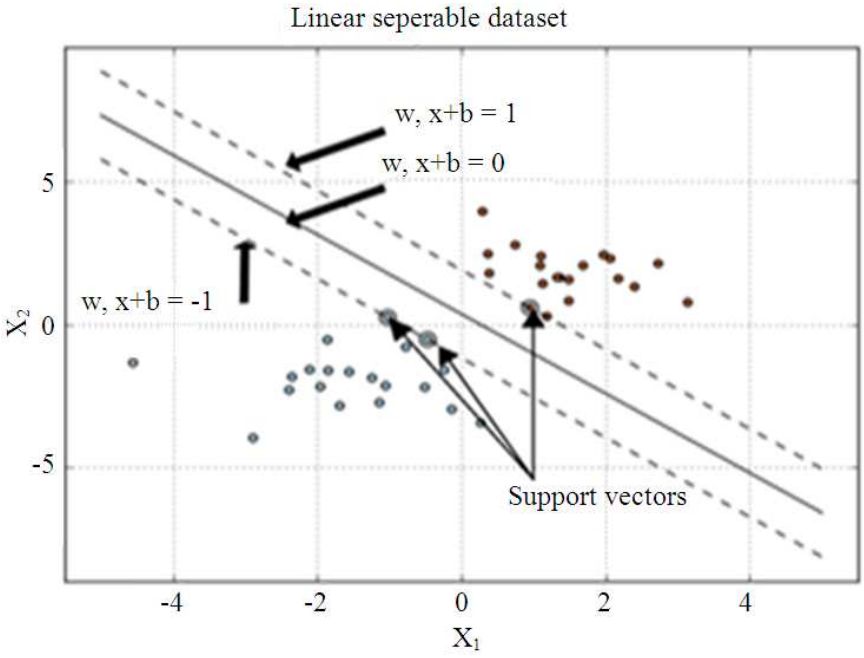

Fig. 2. Classification of an SVM with Maximum-margin hyper-plane trained with samples from two classes. Support vectors are the samples that are on the margin

1.9. Pitfalls of SVM

- The SVM classification efficiency purely depends on initial selection of parameters
- Parameter dependency make SVM more rigid and inflexible to dynamic environment of cloud
- It is a parametric model that cannot be altered later makes SVM as static with fixed parameters
- Requires testing input to be same in the format as in training else it does not classify

1.10. Artificial Neural Network (ANN)

Artificial Neural Networks (ANN) is systems that are deliberately constructed to make use of some organizational principle resembling those of the human brain. They represent the promising new generation of information processing systems. Neural Networks are good at task such as pattern matching and classification, optimization and data clustering. They have a large number of highly interconnected processing elements called neurons, which usually function in parallel and are organized in regular architectures. The collective behavior of a NN, like a human brain, demonstrates the ability to learn recall and generalize from training patterns or data. NNs are characterized by:

- Patter of interconnection between neurons
- Learning algorithm
- Activation function

In a NN, each neuron is connected to the other neuron by means of directed communication link and with an associated weight. Each has an internal stare called as its activity level. Based on the signal flow direction they are classified as feed forward networks and feedback networks. The block diagram of a neuron is shown **Fig. 3**.

The following are the essential three elements of the neuronal model:

- A set of connecting links called synapses; each of the synapse is characterized by a weight or strength of its own. A signal x_j at the input of synapses j connected to the neuron k is multiplied by weight w_{kj}
- An added that performs summation of the input signals
- An activation function to limit the amplitude of the neuron

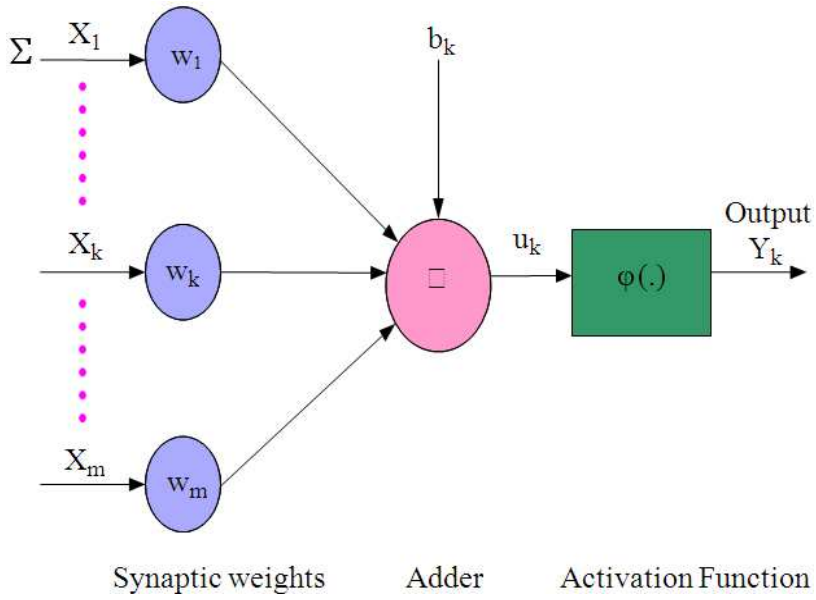

Fig. 3. Model of a neuron

1.11. Design of Neural Network

The decision making problem for choosing an optimal method of resource allocation in cloud can be addressed is via the tuning the coefficients for the constraints. It is helpful in simplifying the cloud agent's architecture associated with saving on both running time and memory. This decision making is also similar to classification problem, for which artificial neural networks exhibited to be a very suitable tool in literature. Artificial Neural networks can be adapted to learn so as to make human-like decisions this would naturally follow any alteration in the data set as the environment changes which eliminates the task of re-tuning the coefficients making its more adaptable for our dynamic cloud environments.

1.12. Feed forward Neural Network

We used a logistic regression model (resource provisioning) to tune the coefficients for the functions $f_1,...,f_4$ for the constraints and evaluate their relative significance. Thus the equivalent conditional probability of the occurrence of the job to be offered is:

$$\hat{y} = P(\text{decision} = 1 | w) = g(w^T f) \quad (14)$$

$$g(a) = e^a / 1 + e^a \quad (15)$$

where, g represents the logistic function which is estimated at activation a. Then, w denote weight vector and whereas f denote the column vector of the importance functions: $f^T = [f_1,\ldots,f_5]$. Then the "decision" is generated according to the logistic regression model.

The weight vector *w* can be adapted using Feed Forward Neural Network (FFNN) topology. In the simplest case there is one input layer and one output logistic layer. Ii is equivalent to the generalized linear regression model with logistic function. The estimated weights satisfy Equation (16):

$$\sum_i w_i = 1,\ 0 \le w_i \le 1 \quad (16)$$

The linear combination of weights with inputs $f_1,...,f_4$ is a monotone function of conditional probability, as shown in Equation (14) and (15), so the conditional probability of job to be offered can be monitored through the changing of the combination of weights with inputs $f_1,...,f_4$. The classification of decision can be achieved through the best threshold with the largest estimated conditional probability from group data. Then the class prediction of an observation x from group y was determined by Equation (17):

$$C(x) = \arg\max_k Pr(x | y = k) \quad (17)$$

To find the best threshold, Receiver Operating Characteristic (ROC) has been used to provide the percentage of detections that are correctly classified and the non-detections which are incorrectly classified. For which, we employed different thresholds with range in [0, 1]. To improve the generalization performance and achieve the best classification, the Multi Layer Perceptron (MLP) with structural learning was employed. An Unidirectional flow of network information in FNN avoiding backward flow Multi-layer perceptron is shown in **Fig. 4**.

1.13. Working of ANN

The same "n" parameters as in SVM are taken here as input and fed in to "n" node of input layer parallel. These input parameters are manipulated in several configurations to a get a better weight matrix that yields a good result with minimal error at output layer. During training, for the given input the weight matrix is adjusted to get the desired result say "1" as Boolean value denoting true in the profit. This updated weight matrix is utilized for testing, during testing input of request is manipulated with weight values and the final result at the output layer decides the success (profit) and failure (loss) of the allocation scenario in accordance to the training details as same as SVM.

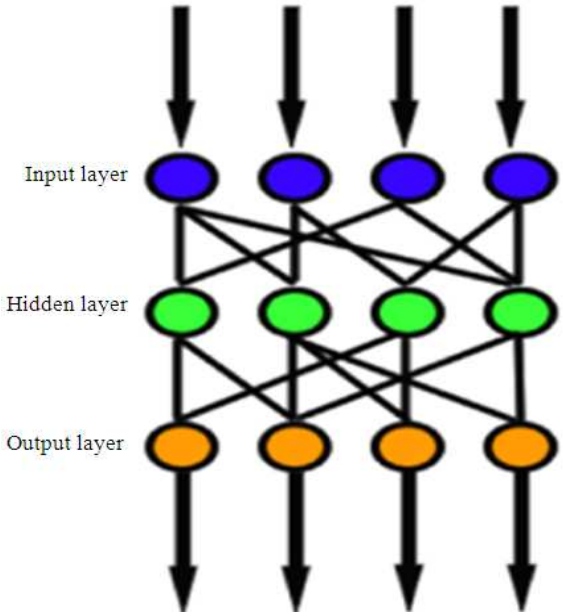

Fig. 4. Unidirectional flow of network information in FNN avoiding backward flow Multi-layer perceptron

1.14. Advantages of ANN over SVM

- When the result is not profitable, it can check for alterations in weight matrix to succeed using back propagation. This makes ANN more dynamic
- Adaptability in both training and testing make it more appropriate for cloud environments
- ANN is non-parametric model containing group of hidden layer based on the input feature and does not restrict the form of the input in testing to be same as during training
- Online training of neural networks is very simple compared to online SVM fitting
- Also in recent years there is a collection of novel algorithms for training neural networks with many layers in sophisticated ways
- It is semi supervised making ANN more efficient than SVM

2. MATERIALS AND METHODS

We used CloudSim as a Cloud environment simulator and implement our resource allocation techniques within this environment. We observe the performance of the proposed methodology over the existing SLA based model in two perspectives namely the user and the resource provider. In the perspective of users, we examine number of requests are accepted and the fastness at which user requests are processed (called average response time). In SaaS provider's perspective, we analyze amount of profit they gain and number of VMs that get initiated. Thus, we use four performance measurement metrics namely Total profit measured in $, Average request response time measured in ms, Total number of initiated VMs. We examine our technique with the total of 500 users.

3. RESULTS AND DISCUSSION

The following **Fig. 5** shows that the Artificial Neural Network (ANN) achieves the highest profit (maximum 15% more than SLA model based and SVM) by accepting (45%) more users and initiating the least number of VMs (19% less than SLA model based, 28% less than SVM) when arrival rate is increases from 100 to 500. This is because ANN accept users with existing machines with penalty delay. In the same scenario, SLA model based and SVM achieve similar profit, but SVM accepts 4% more requests with 13% more VMs than SLA model based. Therefore, in this scenario ANN is the best choice for a SaaS provider. On the other hand, when arrival rate is very large and the number of VM is limited, SVM is a better choice compared to SLA model based because although it provides similar profit as SLA model based, it accepts more requests, leading to market share expanding.

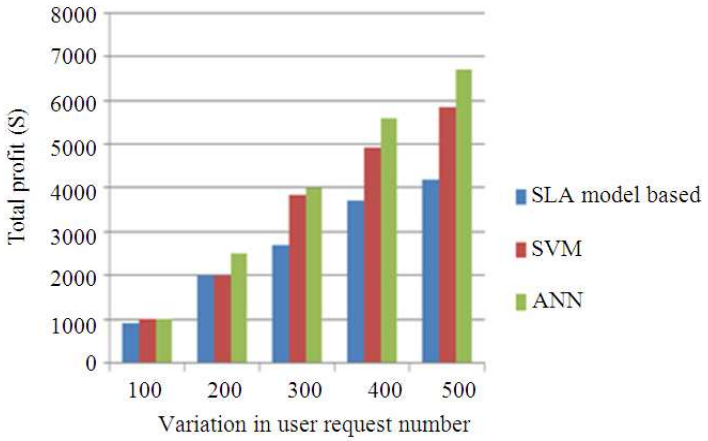

Fig. 5. Total profit

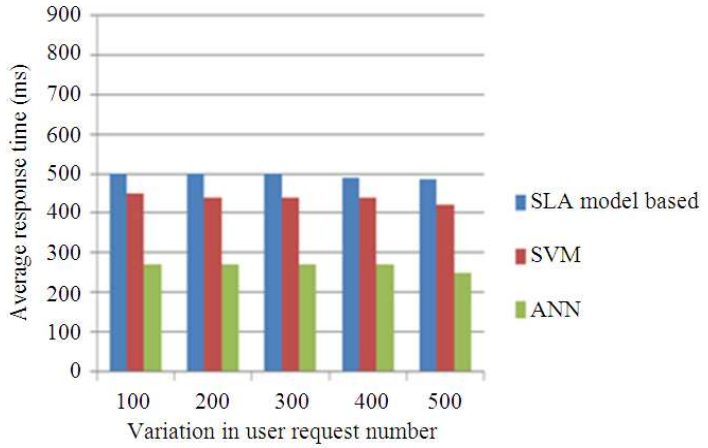

Fig. 6. Average response time

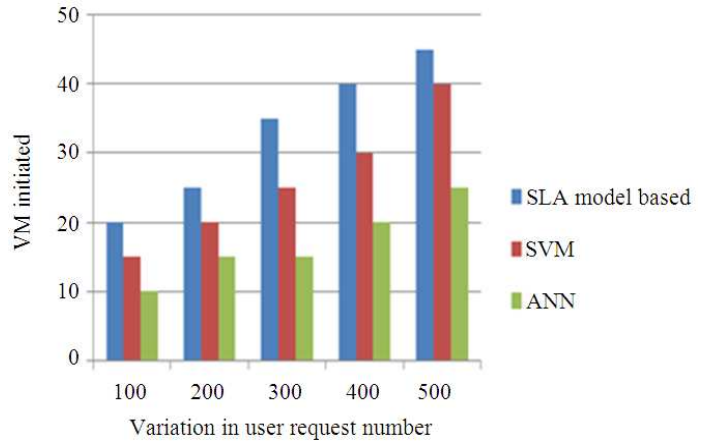

Fig. 7. Number of initiated VM'S

The following **Fig. 6** shows that the ANN achieves in the smallest response time and accepted more number of users with less number of VMs. When the arrival rate is higher, the difference between response time from ANN and its next competitor SVM is twice of ANN. SLA model based and SVM have similar response times. However, there is a drastic increase in response time when the arrival rate is 500 because more requests are accepted per VM which delays the processing of requests.

We can conclude safely that considering the response time constraints from users perspective, the best choice for a SaaS provider is still the ANN.

The following **Fig. 7** shows ANN initiating the least number of VMs (19% less than SVM, 28% less than SLA model based) when arrival rate is increases from "very small (100)" to "very large (500)". This again confirms the effectiveness of ANN.

4. CONCLUSION

We presented here a dynamically adaptable admission control cum scheduling algorithms for efficient resource allocation to maximize profit and CSL for SaaS providers. Through simulation, we showed that the proposed work well in a different kind of scenarios. Our simulation results show that in average the ANN associated system with reduced SLA violation gives the maximum profit (in average save more than 40% VM cost) among all other techniques that ultimately focus on fastest response time among all other methods say SVM and SLA model based. In future we are aiming to consider SLA negotiation in Cloud computing environments to improve the robustness. Here the SLA negotiation is primarily based on the deadline of the work submitted by user. Hence this can be extended to another dimensional view of considering; bandwidth the user uses for process their task, time slot allocated for the user and memory allocated for user service on the resource of SaaS provider. We also liked to add different type of services and other pricing strategies such as spot pricing to increase the profit of service provider. Furthermore, to investigate the knowledge-based admission control and scheduling for maximizing a SaaS provider's profit is one of our future directions for reducing run-time complexity. Resource provided by SaaS provider could be optimized for the better resource allocation scenarios in cloud.

5. REFERENCES

Buyya, R., R. Ranjan and R.N. Calheiros, 2010. InterCloud: Utility-oriented federation of cloud computing environments for scaling of application services. Proceedings of the 10th international conference on Algorithms and Architectures for Parallel Processing, May 21-23, Springer Berlin Heidelberg, Busan, Korea, pp: 13-31. DOI: 10.1007/978-3-642-13119-6_2

Calheiros, R.N., R. Ranjan, A. Beloglazov, C.A.F. De Rose and R. Buyya, 2011. CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Soft. Pract. Experience, 41: 23-50. DOI: 10.1002/spe.995

Catak, O.F. and M.E. Balaban, 2012. CloudSVM: Training an SVM classifier in cloud computing systems. Proceedings of the International Conference on Pervasive Computing and the Networked World, Nov. 28-30, Springer Berlin Heidelberg, Istanbul, Turkey, pp: 57-68. DOI: 10.1007/978-3-642-37015-1_6

Jaideep, D., N. Maheshwari and V. Varma, 2010. Learning based opportunistic admission control algorithm for MapReduce as a service. Proceedings of the 3rd India Software Engineering Conference, Feb. 25-27, ACM Press, New York, USA., pp: 153-160. DOI: 10.1145/1730874.1730903

Lee, Y.C., C. Wang, A.Y. Zomaya and B.B. Zhou, 2010. Profit-driven service request scheduling in clouds. Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, May 17-20, IEEE Xplore Press, Melbourne, Australia, pp: 15-24. DOI: 10.1109/CCGRID.2010.83

Maithili, A., R.V. Kumari and S. Rajamanickam, 2012. Neural networks cum cloud computing approach in diagnosis of cancer. Int. J. Eng. Res. Applic., 2: 428-435.

Reig, G., J. Alonso and J. Guitart, 2010. Deadline constrained prediction of job resource requirements to manage high-level SLAs for SaaS cloud providers. University Politécnica de Catalunya.