Towards an Autonomic Performance Management Approach for a Cloud Broker Environment Using a Decomposition-Coordination Based Methodology

Rajat Mehrotra[a,*], Srishti Srivastava[b], Ioana Banicescu[b], Sherif Abdelwahed[c]

[a] *Applied Innovation Center for Advanced Analytics, Desert Research Institute, USA*
[b] *Department of Computer Science and Engineering, Mississippi State University, USA*
[c] *Department of Electrical and Computer Engineering, Mississippi State University, USA*

Abstract

Efficient resource allocation of computational resources to services is one of the predominant challenges in a cloud computing environment. Furthermore, the advent of cloud brokerage and federated cloud computing systems increases the complexity of cloud resource management. Cloud brokers are considered third party organizations that work as intermediaries between the service providers and the cloud providers. Cloud brokers rent different types of cloud resources from a number of cloud providers and sublet these resources to the requesting service providers. In this paper, an autonomic performance management approach is introduced that provides dynamic resource allocation capabilities for deploying a set of services over a federated cloud computing infrastructure by considering the availability as well as the demand of the cloud computing resources. A distributed control based approach is used for providing autonomic computing features to the proposed framework via a feedback-based control loop. This distributed control based approach is developed using one of the decomposition-coordination methodology, named interaction balance, for interactive bidding of cloud computing resources. The primary goals of the proposed approach are to maintain the service level agreements, maximize the profit, and minimize the operating cost for the service providers and the cloud broker. The application of interaction balance methodology and prioritization of profit maximization for the cloud broker and the service providers during resource allocation are novel contributions of the proposed approach.

Keywords: Decomposition-Coordination, Interaction Balance, Autonomic Computing, Cloud Broker, Cloud Computing.

1. Introduction

Cloud computing is an active area of research that shows an increasing trend towards hosting services in a cloud computing environment in order to eliminate the need for setting up a physical infrastructure, decrease the launching time for a service, provide on demand availability of computing resources, and transfer the resource allocation and management responsibility to the cloud providers, such as Google Apps [1], Amazon Web Services [2], IBM Cloud [3], and Microsoft Windows Azure [4]. Flexibility, reliability, and scalability are prominent features of any

[*] Corresponding author: Tel.: +1 775-673-7609; fax: +1 775-673-7485.

Email addresses: rajat@dri.edu (Rajat Mehrotra), srishti@hpc.msstate.edu (Srishti Srivastava), ioana@cse.msstate.edu (Ioana Banicescu), sherif@ece.msstate.edu (Sherif Abdelwahed)

cloud computing system. Recent research has been focused on capitalizing on these properties of such systems while optimizing other important aspects, such as availability, security, and affordability. In general, a cloud computing system is comprised of service providers and cloud resource providers. The cloud providers allocate computational resources to the service providers for the deployment of the associated services. The cloud providers deploy services on multiple computing nodes in accordance with the service level agreements (SLAs) that are negotiated between the cloud provider and the service provider for a given system resource availability and a specified pay-per-use. In general, the designers of service deployment schemes consider multi-dimensional objectives which are often mutually conflicting in nature. These objectives may include operating costs, SLAs, configuration constraints, resource utilization, availability of the resources, and others. If the deployment scheme is too optimistic (with an underestimation of the resource requirement), it may result in excessive scarcity of resources, leading to SLA violations. However, in the case of a pessimistic scheme (overestimation of the resource requirement), the deployment may be less efficient with respect to resources, and may result in an increased operating cost for the cloud operation without an increased revenue [5]. Moreover, these deployment schemes are applicable only to the specific application domain and operating conditions. Therefore, the underlying system design process has limited utility, reliability, and scope. According to a survey [6], only 26% of the organizations reported improvement in the performance of their applications in a cloud computing environment. The primary reason for this decreased application performance is that these applications (and the underlying services) require complex configurations of computing, network, and storage resources. Currently, SLAs between a service provider and a cloud provider cover only resource availability aspects of a cloud computing infrastructure. There is no agreement specifying the application level performance or quality of service (QoS) that the given resources are able to provide to the applications and their underlying services. A vast number of emerging cloud providers have the capability to address all the requirements of an application or a service provider. The choice of an appropriate cloud provider is a challenging process that calls for an optimized aggregation, integration, and customization of the relevant cloud resources.

Recently, the use of a third party entity, called a *Cloud Broker*, has become popular for selecting the appropriate cloud provider in the interest of service providers, for deployment of their services either in a single or in a federated cloud infrastructure [7, 8]. In other words, the cloud brokers work as intermediaries between cloud providers and service providers to select the appropriate cloud provider, negotiate the contract conditions, and facilitate the deployment of a service in the federated cloud infrastructure. In more advanced cases, a cloud broker can rent the resources from multiple cloud providers and host a service in multiple cloud providers; then, a cloud broker can charge the service provider for these cloud resources at a higher cost than the actual cost paid to the cloud providers. The business and profit models for the cloud brokers are still evolving and research in this area is still ongoing.

This paper considers a federated cloud computing environment where a cloud broker has the ability to integrate resources from more than one cloud provider to support several service providers [8]. These service providers are expected to host web-based services, which are accessed by users over web (http) interface. The cloud broker pays the usage (base) cost of the cloud resources to the cloud provider, and charges the service provider for these resources at some premium on the base cost. In this scenario, the cloud broker and the service provider(s) try to maximize their respective profits, such that a service provider aims to satisfy the service level agreements (SLAs) of the services that it hosts by using the minimum amount of hardware resources from the cloud broker, whereas the cloud broker improves its profits by leasing maximum percentage of computing resources to the service providers. These objectives are conflicting and contradictory to each other. In addition, due to unpredictable variations in the computing environment, the arrival rate of incoming web requests towards the deployed services may exceed its expected value, making it necessary for the service providers to request additional resources from the cloud broker. The service providers must negotiate for these resources with the cloud broker using some pricing scheme to maintain SLAs of the services while minimizing the cost of the utilized cloud resources. Moreover, the cloud broker may be hosting multiple services that may be requesting additional resources at the same time, leading to a competition among the service providers for obtaining these resources, which are limited in number or quantity. This paper introduces a novel negotiation approach between the cloud broker and the various service providers to compute an optimized allocation of cloud resources for the services. The proposed approach is a proof of concept using one of the decomposition-coordination methodology, named interaction balance, for solving the negotiation problem between the cloud broker and the service providers. This paper presents a step towards a model-based autonomous deployment of robust, adaptive, and reliable services in a cloud computing environment. The proposed approach is autonomic for maintaining SLAs of the services (self-optimizing) by dynamic resource allocation and for negotiating pricing to maintain the profit of both the cloud broker and the service providers. A few initial results were presented in an earlier version of this paper [9].

This paper is organized as follows. Related research efforts made by other research groups are presented in Section 2. The proposed performance management approach is introduced in Section 3. The application of the proposed approach in a typical cloud infrastructure is demonstrated in Section 4 and the benefits of the approach are highlighted in Section 5. Finally, conclusions and future work are presented in Section 6.

2. Related Work

2.1. Resource Allocation in a Cloud Computing Environment

Efficient resource allocation is one of the most challenging problems faced by the cloud infrastructure as a service (IaaS) providers. Academia and research communities have proposed several new technologies for dynamic resource allocation in this domain to maintain the SLAs. A dynamic resource allocation method is developed by considering the SLAs between the user and the Software as a Service (SaaS) provider while allocating resources [10]. The authors ensure that SaaS providers are able to manage the dynamic change in customer's requests, mapping customer requests to infrastructure level parameters, and handling the heterogeneity of the Virtual Machines (VMs). This approach also considers the customers' QoS parameters, such as response time, and the infrastructure level parameters such as service initiation time. However, the burden of evaluating SLAs between the user and the SaaS providers may become a bottleneck for the IaaS cloud provider when the number of SaaS providers is considerably large. Another prominent approach of resource allocation that has been investigated by researchers, is priority driven resource allocation [11, 12]. These approaches are classified in two categories: (i) user priority based and (ii) resource priority based. These approaches only consider resource allocation to a single service provider for solving the load balancing problem.

A neural network based resource allocation is introduced in [13], where authors focus on maximizing the resource utilization via an efficient resource allocation strategy provided by the genetic algorithms. However, this approach targets a system where the resources are not scarce. There is no competition among the users for obtaining a set of required resources. Another approach uses genetic algorithms to find the optimal resource allocation and assigns resources to the clients or users based on the outcome of the genetic algorithm [14].

Recently, researchers have focused on cloud resource allocation by applying auctioning schemes to the SaaS providers. In these auctioning schemes, the cloud IaaS providers accept requests from SaaS providers and perform an auction of the cloud resources. The highest bidder will be allocated the set of auctioned resources. To deal with such complexities of the resource allocation problem in a dynamic and evolutionary environment, a number of researchers have focused on a study of game theoretical approaches [15]. Game theory-based resource allocation mechanisms have received a considerable amount of attention in cloud computing to solve the optimization problem of resource allocation. However, the recent survey shows that these techniques do not take into consideration essential parameters such as, fairness, resource availability, service deadline and execution efficiency, resource reliability, etc. More often, the game-theoretic approaches focus on solving the cost optimization problems. A combinatorial auction-based mechanism has also been investigated for the allocation and pricing of VM instances in cloud computing platforms [16]. This approach uses three different schemes, namely, fixed-price scheme, linear programming, and a greedy scheme. However, this approach only considers maximizing the user gains as a single constraint that limits the allocation of each type of VM to a pre-determined value.

2.2. Significance of Cloud Brokers

The approaches discussed in the previous subsection present solutions to address the challenges of efficient resource management between cloud provider and service provider(s). However, these approaches do not address the resource management problems in an enterprise framework, which makes use of services provided by various cloud providers to fulfill SLAs. The main advantage of using the cloud brokers in this scenario, is their ability to integrate more than one cloud provider. These capabilities need to be exposed to enable the fulfillment of all orchestration requirements. According to recent literature, due to the increase in the demand for a federated cloud computing framework, efficient management and operation of cloud computing environment is the most prominent requirement [8], [17]. Cloud brokers are the most promising solution available to deal with the complexities of a federated cloud environment. Therefore, there is a need for applying efficient resource allocation and management in a cloud broker model.

Recently, researchers have proposed various solutions to address the federated cloud management problem at different levels. One of these solutions proposed managing the information of a large number of cloud service providers via a unique indexing technique [18]. The other solution proposed enabling the cloud-computing services broker to use derivative contracts in combination to reliably providing cheaper resources to the consumer and predicting future usage [17]. Another approach proposed a novel secure sharing mechanism for a secure cloud bursting and aggregation operation such that the cloud resources are shared in a secured manner among different cloud environments [19]. However, none of these research directions address the issues of dynamic resource allocation and performance management of hosted services in a cloud broker model. In addition, these resource allocation and performance management issues become more critical in the case of large number of hosted services with limited amount of cloud resources. In these situations, maintaining the SLAs of the services' becomes crucial and requires an autonomic performance management approach which can dynamically reallocate the cloud resources among services while maximizing the profitability of the cloud broker.

2.3. Large Scale Control-Based Performance Management Approaches

Large scale control-based methods are promising ways to automate certain system management tasks encountered in distributed computing systems. Algorithms have been developed for optimal control of large scale systems by decomposing the large systems into a number of interconnected subsystems. Thus, the system wide optimization problem is also divided into a number of subsystem optimization problems. These subsystems coordinate with each other through a coordinator using interaction inputs, and thus achieve the system wide performance objectives. These interaction inputs are applied to each subsystem in the form of constraints. These "decomposition and coordination" strategies are primarily implemented in two ways: Interaction Balance (Goal Coordination) and Interaction Prediction (Model Coordination) [20]. Both of these approaches have been applied successfully to a number of large scale systems, where subsystems are "coupled with each other in both system dynamics and system wide performance objectives" [20].

Furthermore, application performance issues in cloud broker environment can be addressed by service providers through developing service specific controllers that can manage the requirements of a service for computing, network, and storage resources. These service level controllers can be designed, developed, and deployed by the service providers at each service independent of the cloud level controllers. The cloud level controllers are deployed by the cloud brokers (or cloud providers) to ensure resource availability and minimum downtime for the deployed service as negotiated in SLAs with the service providers. Thus service providers can choose and customize their own control policies inside service level controllers according to the service requirements.

Contribution of the paper: In this paper, a distributed control algorithm-based autonomic approach is developed for performance management of services hosted in distributed cloud broker environments. The proposed distributed algorithm utilizes the interaction balance based approach, where each service is *decoupled* from other services with respect to system dynamics, while *coupled* in terms of overall deployment wide operating cost functions by limited amount of system resources in a cloud broker infrastructure. In this management approach, we first dynamically allocate the computational resources to all of the service providers through an interaction balance based approach, and then the service level controllers utilize the allocated resources for service deployment to maintain the SLAs their respective services. Important results from prior work on the distributed control of large scale systems [20, 21, 22, 23] are utilized in this paper, where the main idea is of cooperation among multiple independent services to optimize a global cost function under certain constraints (limitation of resources) by independently optimizing the local cost function at each service. The proposed approach is autonomic because it does not require any manual intervention after initial deployment of the services in a cloud broker environment to dynamically allocate the resources or to negotiate the pricing of available computing resources. Based on our survey, until now, there is no published research or study about applying the interaction balance method for optimal control of service providers and cloud broker (or cloud provider) profit maximization in cloud computing systems.

3. A Distributed Control-Based Performance Management Approach Using the Interaction Balance Principle

In this paper, a single cloud broker interacts with multiple service providers during auctioning of the available cloud computing resources. In this situation, the cloud broker broadcasts the initial unit price β_{ini} of the resource per unit time to the service providers. Service providers solve their control problem by using their own utility function, compute the optimal value of a cloud resource, and request the resource from the cloud broker. If the total amount of resource requested by the service providers is greater than the available resource, the cloud broker increases the unit price of the resource, otherwise the cloud broker reduces the price to encourage the service providers to reserve a higher amount (or number) of resources. The cloud broker sends the updated unit price to the service providers; the service providers again compute the required amount of resource on the updated price. This cycle continues until either the cloud broker sub-lets all of its resources (or very small percentage ϵ left unassigned), or the unit price becomes lower than the minimum (or threshold) price β_{min}, or the service providers refuse to pay the current price considered as β_{max}. Here, $\beta_{min} \leq \beta_{ini} \leq \beta(k) \leq \beta_{max}$, where $\beta(k)$ is the unit price of the cloud resources during time sample k. A detailed list of symbols used in this paper is presented on Page 7.

In this scenario, N service providers are interested to deploy their services in the cloud infrastructure through a cloud broker as shown in Figure 1. Each of the service provider is expected to deploy only one service. All of these services are expected to be web-based services, which are accessed by users over web (http) interface. The proposed management approach is based on the decomposition of the global profit maximization problem of cloud broker in to N sub-problems, which are further solved by each service providers for their own services. These service providers are contesting for total cloud resources $U(k)$ at a particular time instance k. Therefore, in case of limited resources, the resources acquired by the i-th service provider will impact the resource available to all the other service providers j (where $j \neq i$) because they are competing for the same type of cloud resource. The state dynamics at each service provider i (or service i) can be described by the following set of equations.

$$\hat{q}_i(k+1) = \left[q_i(k) + \hat{\omega}_i(k)) - \frac{u_i(k)T}{\hat{c}_i(k)} \right]^+ \qquad (1)$$

$$\hat{r}_i(k+1) = (1 + \hat{q}_i(k+1)) \; \frac{\hat{c}_i(k)}{u_i(k+1)} \qquad (2)$$

$$u_i(k) = \alpha_i(k)\, U(k) \qquad (3)$$

where $[a]^+ = \max(0, a)$, $q_i(k)$ is the local service queue (pending web requests) size of the service i, and $\hat{\omega}_i(k)$ is the expected arrival rate of web requests at the service. $\hat{q}_i(k+1)$ is the expected queue level of the service i, $\hat{r}_i(k+1)$ is the expected response time, T is sampling interval, and $u_i(k)$ is the amount of computational resources (CPU core frequency) acquired by the i-th service provider (for service i) as fraction $\alpha_i(k)$ from the total available cloud resource $U(k)$. $\hat{c}_i(k)$ is the predicted average service time per request at one unit of computational resource (1 GHz. frequency) at service i.

Symbol	**Description**
N	Total number of service providers
H	Size of look ahead horizon used in control algorithm
β_{min}	Minimum unit price of the cloud resource
β_{ini}	Initial unit price of the cloud resource
β_{max}	Maximum unit price of the cloud resource
$\beta(k)$	Unit price of the cloud resource during time sample k
i	indices for identifying the service provider. Here, $i \in (1, N)$
k	indices for time sample. Here, $k \in (1, H)$
$q_i(k)$	Local service queue at i-th service provider during time sample k
$\hat{q}_i(k)$	Expected local service queue at i-th service provider during time sample k
$r_i(k)$	Response time at i-th service provider during time sample k
$\hat{r}_i(k)$	Expected response time at i-th service provider during time sample k
$\omega_i(k)$	Arrival rate of incoming web requests towards i-th service provider during time sample k
$\hat{\omega}_i(k)$	Expected arrival rate of incoming web requests towards i-th service provider during time sample k
$u_i(k)$	Amount of computational resources acquired by the i-th service provider during time sample k
$U(k)$	Total amount of available cloud resource during time sample k
T	Sampling interval
$\hat{c}_i(k)$	Predicted average service time per request at one unit of computational resource at i-th service provider during time sample k
$\alpha_i(k)$	Fraction of cloud resource utilized by i-th service provider during time sample k from the total available cloud resource $U(k)$
$\alpha_j^*(k)$	Fractions of the cloud resources acquired by other service providers j ($\forall j$, $j \neq i$)
q_i^s	Recommended local service queue at i-th service provider
r_i^s	Recommended response time at i-th service provider to maintain SLAs
$E(u_i(k))$	Cost of acquiring $u_i(k)$ amount of cloud resource at i-th service provider during time sample k
$P(k)$	Profitability of the cloud broker during time sample k
$J_i(k)$	Operating cost of the i-th service during time sample k
A_i, B_i, C_i	user defined norm weights for i-th service in the cost $J_i(k)$
$L_i(k)$	The Lagrangian of service provider i during time sample k
$\beta(k)$	Pricing matrix for unit cloud resources during time sample k. Here $\beta \in R^{NH}$
R^{NH}	Matrix of real numbers with N rows and H columns
β_i	Price vector corresponding to the service provider i. Here $\beta_i \in R^H$
R^H	Vector of real numbers with H elements
$\beta_i^l(k)$	Price vector corresponding to the service provider i during l-th iteration between cloud broker and i-th service provider
ϵ	Very small percentage of cloud resources that can be left unassigned
e	Interaction error computed at cloud broker
ξ^l	Step length at l-th iteration between cloud broker and service provider
$d^l(k)$	Search direction at l-th iteration during time sample k
$\sigma^{l+1}(k)$	Ratio of interaction error between l and $(l+1)$ iteration during time sample k

Table : List of Symbols.

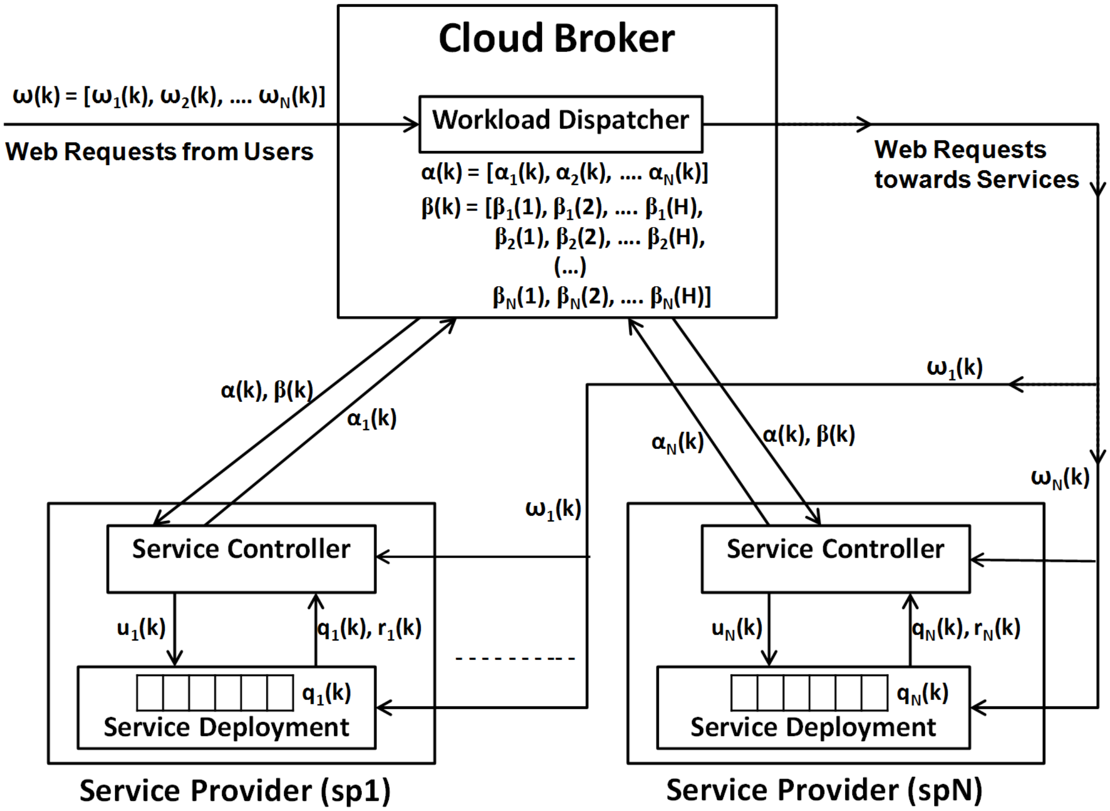


Figure 1: A Distributed Control Structure for Resource Allocation and Performance Management

These equations (1, 2, and 3) represent the aggregate performance behavior of the i-th service with respect to the available computing resources $u_i(k)$, incoming workload $\omega_i(k)$, and existing queue size $q_i(k)$. The actual i-th service deployment problem on multiple computing nodes, and the resource allocation map among these computing nodes is not considered in this paper. However, this problem has already been addressed by the authors previously in a very generic manner [22, 23]. Similarly, the problem of allocating the desired resources to each of the service providers by the cloud broker via an appropriate placement map, is also not addressed in this paper. This paper considers the computational resources as a chunk of resources, while solving specific performance management problems.

The operating cost of the service provider i (or service i) for a look ahead horizon of H steps, $J_i(k)$ includes SLA violation penalty and renting cost of the cloud resources as expressed in Equation 4. The renting cost of the cloud resources depends on its computational resource $u_i(k)$ (CPU core frequency) and represented as $E(u_i(k))$. A_i, B_i, and C_i are user defined norm weights for i-th service. To include the effect of the coordinated goal, the operating cost J_i at the service provider i is indirectly coupled with all the other service providers through limited cloud resources as shown in Equation 5. Therefore, changes in $u_i(k)$ at the service provider i affects the cost function $J_i(k)$ as well as $J_j(k)$, where $j \neq i$.

$$J_i(k) = \sum_{k=1}^{H} \left\|q_i(k+1) - q_i^s\right\|_{A_i} + \left\|r_i(k+1) - r_i^s\right\|_{B_i} + \|E(u_i(k))\|_{C_i} \tag{4}$$

$$u_i(k) = \alpha_i(k)\, U(k) \tag{5}$$

$$P(k) = \beta(k) \sum_{i=1}^{N} u_i(k) - \beta_{min}(k) U(k) \tag{6}$$

Here, $k = 1, 2, .., H$ represents the sampling instances in the trajectory of the system operation. q_i^s and r_i^s are the recommended queue size and response time, respectively. Recommended response time r_i^s is chosen per the SLAs negotiated between the i-th service provider and the end user of i-th service. According to this quadratic cost function in Equation 4, the service providers are penalized for the number of the requests remaining in the queue $q_i(k)$ and the average response time $r_i(k)$ observed at the service. Therefore, q_i^s is generally set to *zero* for the complete depletion of the queue, and r_i^s is set according to the SLAs. Also, $P(k)$ (in Equation 6) represents the profitability of the cloud broker for acquiring the cloud resources, and then subletting them out to the service providers. $U(k)$ represents the total resources offered for auction at time instance k.

As described previously, the profitability of the cloud broker will be maximized if it allocates the maximum (close to 100%) available resources to the service providers. In this process, the cloud broker may have to decrease the unit price $\beta(k)$ (from the initial unit price β_{ini}) of the cloud resources too in order to encourage the service providers for extra resource allocation. Otherwise, the cloud broker can also increase the unit price of a resource (from the initial unit price β_{ini}) when there are not enough resources available to satisfy the requirement of all the service providers. The total revenue of the cloud broker from Equation 6 can be maximized by having the perfect balance of unit price $\beta(k)$ with total allocated cloud resource $\sum_{i=1}^{N} u_i(k)$.

Therefore, the overall cloud resource optimization problem is : *find the optimal value for unit price $\beta(k)$ of cloud resources and resource fraction $\alpha_i(k)$ at each service provider, such that the cloud broker can maximize its profits while* *service providers maintain their own profitability, SLAs, and operational constraints.*

3.1. Problem Decomposition

The cloud broker considered here hosts N service providers, which are coupled only through the cloud resources $u_i(k)$ and cloud broker level cost function $J(k)$. The cloud broker level cost function $J(k)$ is the sum of the cost functions related to each service provider i as $J_i(k)$ in Equation 4. For a proper decomposition, an interaction variable $Z_i(k)$ must be defined at the service provider i to represent the effect of the service provider dynamics on the global cost. $Z_i(k)$ is chosen as the sum of the fractions α_j^* of the cloud resources acquired by other service providers j ($\forall j$, $j \neq i$) as follows. That is, $Z_i(k) = \sum_{j\neq i}^{N} \alpha_j^*(k)$. The constraint at the service provider i is $\alpha_i(k) = 1 - Z_i(k)$ or $\sum_{j=1,j\neq i}^{N} \alpha_j^*(k) + \alpha_i(k) = 1$.

This constraint minimization problem of minimizing cost function $J_i(k)$ at each service provider under the constraint of $\sum_{j=1,j\neq i}^{N} \alpha_j^*(k) + \alpha_i(k) = 1$ can be solved by using Lagrange Multiplier theorem [20]. According to Lagrange Multiplier theorem, local maxima and minima of cost function $J_i(k)$ subject to constraints ($\sum_{j=1,j\neq i}^{N} \alpha_j^*(k) + \alpha_i(k) = 1$) can be calculated by introducing another new variable β_i called Lagrange multiplier, and study the Lagrangian L_i at each service provider. By introducing Lagrangian multipliers, the constrained optimization problem is converted to unconstrained problem, which can be solved at each service provider. The Lagrange multiplier $\beta_i(k)$ introduces a penalty factor in addition to service provider level cost function $J_i(k)$ when missing the constraint expressed as $(1 - \alpha_i(k) - \sum_{j\neq i}^{N} \alpha_j^*(k))$ in $L_i(k)$. The Lagrangian L_i of service provider i can be represented as follows.

$$L_i(k) = J_i(k) + \sum_{k=1}^{H} \beta_i(k)(1 - \alpha_i(k) - \sum_{j\neq i}^{N} \alpha_j^*(k)) \tag{7}$$

where $\beta_i(k) \in R^H$ is the price vector corresponding to the service provider i that is extracted from the Lagrange multiplier vector $\beta(k)$ received from the cloud broker (see Figure 1), where $\beta(k) \in R^{NH}$. Here R^H denotes a vector of real numbers with H elements, and R^{NH} denotes a matrix of real numbers with N rows and H columns. The price vector $\beta_i(k)$ is chosen as Lagrange multiplier here because it reflects the change in price with respect to the constraints of total acquired resources ($\sum_{i=1}^{N} \alpha_i^*(k)$) by the service providers. If service providers miss the constraint negatively ($1 < \sum_{i=1}^{N} \alpha_i^*(k)$), the Lagrange multiplier β will increase which is equivalent to increasing the price of the resources if the demand is higher than the availability. Similarly, when constraint is missed positively ($1 > \sum_{i=1}^{N} \alpha_i^*(k)$), the Lagrange multiplier β will decrease, which is the same as decreasing the price of the resources if the demand is lower than the availability. This decrease in price will encourage maximum allocation of the available computing resources.

The Lagrangian $L(k)$ for the cost function $J(k)$ can be represented as sum of $L_i(k)$: $L(k) = \sum_{i=1}^{N} L_i(k)$. The overall problem of minimizing cost function J can be decomposed in to N first level problems of minimizing L_i, such that Equation 1 is satisfied with $q_i(1) = 0$. The problem at the cloud broker level can be expressed as updating the value of β, so that the interaction error can become less than a pre-defined small value ϵ, where ϵ can be chosen in percentage of the resource not assigned to any of the service providers (5%). It indicates that cloud broker will keep running the auction until at least 95% of the available resources ($U(k)$) are desired by the service providers. Once the auction is complete, all the service providers will receive resources at the current per unit price $\beta(k)$.

3.2. Service Provider Level Control

At the service provider level, the Lagrangian L_i is minimized using service dynamics (Equation 2 and 3) with the cloud resource as control input $u_i(k)$. We create a uniform discretization F_i for the resource fraction α_i. For example, $F_i = [0.05, 0.1, ..., 0.95, 1.0]$. The optimal value of the control inputs $u_i(k)$ is computed that minimizes $L_i(k)$ by using the following steps.

1. Use $\beta_i^l(k)$ and $\alpha_j^{*(l)}(k)$ (where $j \neq i$) as received from the cloud broker to compute the optimal sequence of $(\alpha_i^{*(l)}(k))$ over the horizon $k \in [1, H]$ by using Algorithm presented on Page 11, which minimizes the Lagrangian $L_i(k)$ in Equation 7 through a tree search method [24]. Here, l indicates the iteration instance between the service provider and the cloud broker within time sample k.
2. Forward the optimal values of $\alpha_i^{*(l)}$ to the cloud broker.

Algorithm A Predictive Control Algorithm at the Service Provider-i: $PredictiveControl_i(k)$

```
Input: Service Provider queue $q_i(k)$, Prediction Horizon $H$,
Input: Total amount of cloud resource available for auction $U(k)$,
Input: $\beta$ and $\alpha_j^{*(l)}(k)$ received from the cloud broker,
Input: $\hat{\omega}_i(k)$ estimated by the traffic estimator,
Input: Fraction Set at the service provider $i$, $F_i = [F_{i1}, F_{i2}, ..., F_{iR}]$
Input: Resource share expected at the service provider $i$, $F_i = \alpha_i^l = [\alpha_i^l(1), \alpha_i^l(2), .., \alpha_i^l(H)]$
service provider state $x_i(k) = [q_i(k)\ r_i(k)]$
$s_k := x_i(k), Cost_i(k) = 0$
for all fraction set $f \in F_i$ do
    $\alpha_i^l(1) = \alpha_i^l(2) = .. = \alpha_i^l(H) = f$ /* Same resource share at each step */
    for all $k$ within prediction horizon of depth $H$ do
        $s_{k+1} := \phi$
        for all $x \in s_k$ do
            Compute $\hat{q}_i(k+1)$ /* using Equation 1 */
            Compute $\hat{r}_i(k+1)$ /* using Equation 2 */
            Compute $L_i(k+1)$ /* using Equation 7 */
            $Cost_i(k+1) := Cost_i(k) + L_i(k+1)$
            $\hat{x}_i = [\hat{q}_i(k+1)\ \hat{r}_i(k+1)]$
            $s_{k+1} := s_{k+1} \cup \{\hat{x}\}$
        end for
        $k := k + 1$
    end for
end for
Find $x_{min} \in s_N$ having minimum $Cost_i(k)$ /* $x_{min}$ is the service state, which has minimum total cost of leading
    from $x_i(k)$ to $x_{min}$ */
Choose $f$ with minimum $L_i(k)$, where $f = \alpha_i^{*(l)} = [\alpha_i^{*(l)}(k), .., \alpha_i^{*(l)}(k+H-1)]$
return $\alpha_i^{*(l)}$
```

3.3. Cloud Broker Level Control

At the cloud broker level, the goal is to update the values of the Lagrange multipliers β to decrease the interaction error e, defined as:

$$e_i^l(k) = 1 - \sum_{j=1}^{N} \alpha_j^{*(l)}(k) \tag{8}$$

$$e^l = \begin{pmatrix} e_1^l & e_2^l & \dots & e_i^l & \dots & e_N^l \end{pmatrix}^T \tag{9}$$

$$e_i^l = \begin{pmatrix} e_i^l(1) & e_i^l(2) & \dots & e_i^l(k) & \dots & e_i^l(H) \end{pmatrix}^T \tag{10}$$

The interaction error vector e is used as a gradient to modify the Lagrange multipliers $\beta(k)$ using the conjugate gradient method [20] per following set of equations:

$$\beta^{(l+1)}(k) = \beta^{(l)}(k) + \xi^l \; d^l(k) \tag{11}$$

Where, ξ^l represents the step length, and d^l the represents search direction. $d^l(k)$ is calculated using following set of equations with $d^0 = e^0$.

$$d^{l+1}(k) = -e^{l+1}(k) + \sigma^{l+1} \; d^l(k) \tag{12}$$

$$\sigma^{l+1} = \frac{||e^{l+1}||}{||e^l||} \tag{13}$$

$|| \cdot ||$ denotes the (Cartesian) ℓ_2-norm. The main steps of the algorithm at the cloud broker level are as follows:

1. Set initial values of the Lagrange multipliers vector β as the initial unit price of a resource (β_{ini}), and forward it to the service provider level controllers.
2. The cloud broker uses the values of α_i^* received from the i-th service provider to calculate the interaction error e using Equation 9.
3. If $||e^l|| \leq \epsilon$, stop and assign the cloud resource to service providers in requested ratio, else go to next step.
4. Calculate the values of the Lagrange multipliers β for the next iteration by using Equation 11, 12, and 13. If calculated value of β is more than the maximum unit price (β_{max}) of the resources, or less than the minimum unit price (β_{min}), stop and assign the requested amount of cloud resources to each service provider. Otherwise, send this updated value of β to the service level controllers for solving the service provider level optimization problem. Increment l and jump to Step 2. This exchange of information is shown in Figure 1.

3.4. Forecasting the Environmental Inputs at each Service Provider

In general, the web services considered in this paper, are deployed in a dynamic and open distributed environment, where the incoming web requests (environmental inputs) are generated from external users (or clients) that cannot be controlled by the services. In addition, these web requests show a cyclic pattern in the arrival rate based upon the service popularity and time of the day [25]. These web requests vary significantly within a short duration of a few minutes. However, this variation in the arrival rate of the incoming web requests can be estimated within certain accuracy using Autoregressive Moving Average (ARIMA) filters [26] or Kalman filters [27], as done in earlier work [28, 29]. In this paper, an ARIMA filter based prediction module, *Traffic Estimator* is developed to estimate the arrival rate of incoming web requests $\hat{\omega}_i(k)$ at each of the service providers. This traffic estimator module is developed

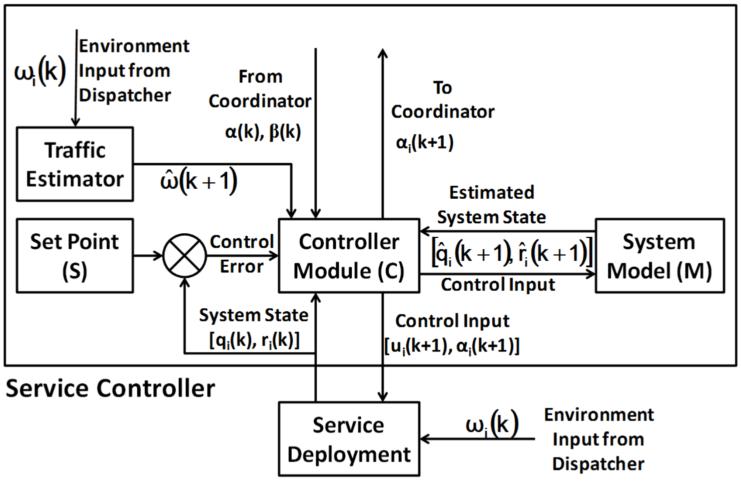


Figure 2: Service Provider Level Control Structure for Computing Optimal Values of the Control Input

by using the similar approach used in [28]. This module is shown in Figure 2. The expected arrival rate $\hat{\omega}_i(k)$ of web requests at the service i can be estimated by equation 14.

$$\hat{\omega}_i(k) = \theta(\omega_i(k-1, r)) = \gamma_{i1}\, \omega_i(k-1) + \gamma_{i2}\, \omega_i(k-2) + (1 - (\gamma_{i1} + \gamma_{i2}))\, \bar{\omega}_i(k-3, r) \tag{14}$$

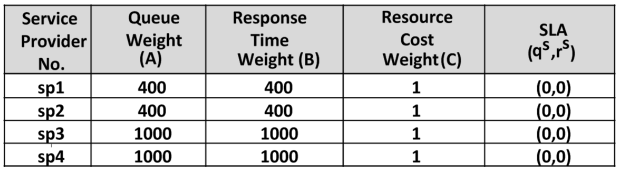
where, γ_{i1} and γ_{i2} are user specified weights on the current and previous arrival rates at service i. $\bar{\omega}(k-3, r)$ represents the average arrival rate of the web requests between time samples $k-3$ and $k-3-r$. This prediction module continuously monitors the arrival rate of the incoming web requests and estimates their future values at each sample.

4. Performance Management of Service Providers in a Cloud Computing Environment

The proposed interaction balance based resource allocation approach is simulated in Matlab for deriving an optimal resource allocation and managing the performance of four service providers that are hosting their respective services in a cloud computing environment using a cloud broker. Details of the simulations are described in the following subsections.

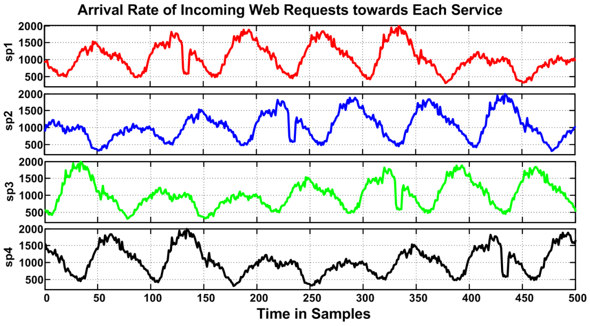
4.1. Service Provider Level Control Setup

The service provider level controller dynamics is shown in Figure 2. Each of the service providers receives incoming web requests $\omega_i(k)$ from the Dispatcher at the cloud broker during time sample k. Each service deployment processes the incoming web requests by using the allocated computational resource $u_i(k)$, which is a fraction $\alpha_i(k)$ of the computational resource available at the cloud broker. The controller module C receives the current system state $[q_i(k), r_i(k)]$ from the service deployment, future workload arrival rate $\hat{\omega}_i(k+1)$ from the traffic estimator, and the Lagrange multipliers β from the cloud broker. Now, controller module (C) uses the service provider model (M) and the available control algorithms to obtain the optimal value of resource share $\alpha_i(k+1)]$ by using Algorithm described on 11. The Set Point (S) of the service deployment is the recommended queue size q_i^s and response time r_i^s. The calculated resource share $\alpha_i(k+1)$ is sent back to the cloud broker to calculate the interaction error e.



Service Provider No.	Queue Weight (A)	Response Time Weight (B)	Resource Cost Weight(C)	SLA (q^S,r^S)
sp1	400	400	1	(0,0)
sp2	400	400	1	(0,0)
sp3	1000	1000	1	(0,0)
sp4	1000	1000	1	(0,0)

(a) The Parameter Values Used for the Simulation Experiment.



(b) Simulated Incoming Web Requests towards Web Services.

Figure 3: Simulation Parameters and Arrival Rate of Web Requests for each Service Provider. q^s and r^s are recommended queue size and response time, respectively.

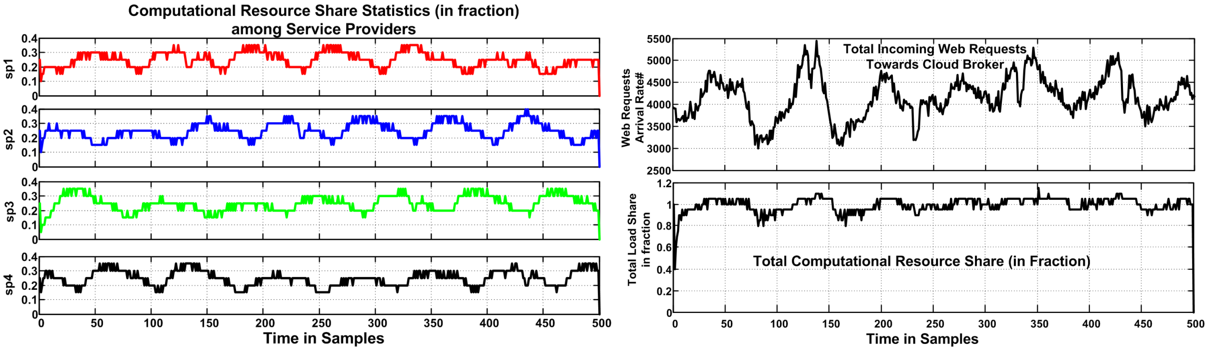
4.2. Simulation Setup

The simulation settings and the coefficients used in the cost function are shown in Figure 3(a) for four service providers $sp1$, $sp2$, $sp3$, and $sp4$. The Service provider $sp3$ and $sp4$ are assigned higher penalty for queue size and response time compared to the service providers $sp1$ and $sp2$. Therefore, the service providers $sp3$ and $sp4$ are expected to choose higher amount of computational resource compared to service providers $sp1$ and $sp2$. All of these service providers are assigned lower penalty for resource cost compared to the queue size and the response time, which will force all of these to focus more on the SLAs (queue size and response time) while calculating the optimal values of computational resources.

During this simulation, different web request workloads (see Figure 3(b)) are generated for each services by utilizing the 1998 Football World Cup [25] traces. The tolerance value ϵ of the interaction error is set to 0.05 and the lookahead horizon H is set to 2. In addition, the minimum (β_{min}), maximum (β_{max}), and initial (β_{ini}) unit price of cloud computing resource is set to 1000, 10000, and 2500 pricing unit, respectively. The total available cloud resources at cloud broker for the simulation is constant as 16 GHz. during the entire simulation. These simulations can also be repeated with higher amount of total cloud resources, different pricing value, and higher rate of incoming web requests for each service.

4.3. Simulation Results

This simulation is conducted to demonstrate the performance of the proposed performance management approach in a cloud computing environment for maximizing the profitability of the cloud broker while managing the SLAs of the service providers in a dynamic environment. During the simulation, dynamic arrival rate of incoming web requests (see Figure 3(b)) are utilized for the service providers that facilitate competition among the service providers to acquire higher amount (or number) of cloud resources during few time samples due to an extremely high rate of incoming web requests. In contrast to this, during a few time samples, the total desired amount of resources is much smaller than the total amount of available resources at the cloud broker due to low arrival rate of incoming web requests towards the services. The results of this simulation are shown in Figure 4, 5, and 6.



(a) Computational Resource Share Statistics at Each Service Provider.

(b) Total Web Requests and Total Computational Resource Share Statistics (in fraction) at Cloud Broker Level.

Figure 4: Computational Resource Sharing Among Service Providers and Total Incoming Web Requests Analysis for the Cloud Broker.

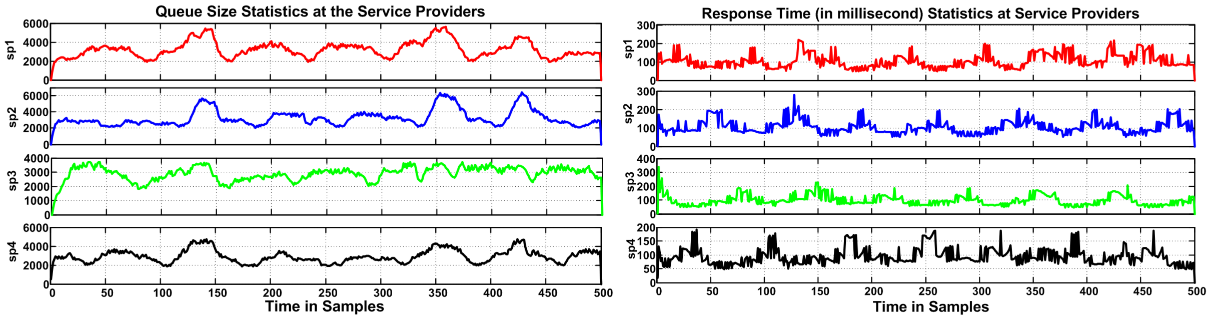
Computational Resource Statistics

Figure 4(a) shows the share of computational resource acquired by each of the service providers to maintain their respective SLAs. These service providers change their resource share based on the intensity of incoming web requests (see Figure 3(b)). Initially, service provider $sp3$ receives a higher number of web requests compared to other service providers, therefore in the beginning, service provider $sp3$ acquires maximum share of computational resources. Similar phenomenon is observed in case of service provider $sp1$, $sp2$, and $sp4$ during other time samples. Furthermore, the resource allocations adapts to the changes in the web request arrival rate by changing the resource distribution on the services to maintain their individual SLAs and maximizing the profitability of the cloud broker simultaneously.

Figure 4(b) shows the total web requests (sum of the web requests of all the service providers) arrived at the cloud broker during a time sample, and the total share of cloud resources acquired by the service providers. Ideally, the total share of workload should always be 1, i.e., cloud broker should be able to change the unit price of the computational resource to facilitate 100% allocation. However, in some scenarios when the total incoming web requests are too low (see Figure 4(b), during time sample 160, 240, etc), the total allocation is lower than 100% (less than 1). In a few time samples, this total resource allocation goes to as minimum as 80% (or 0.8). On the contrary, when the total web requests are at the peak rate, the total share of desired workload increases beyond 1 (100% allocation), because in these situations, the service providers compete for extra computational resources to maintain their SLAs.

Queue Size and Response Time Statistics

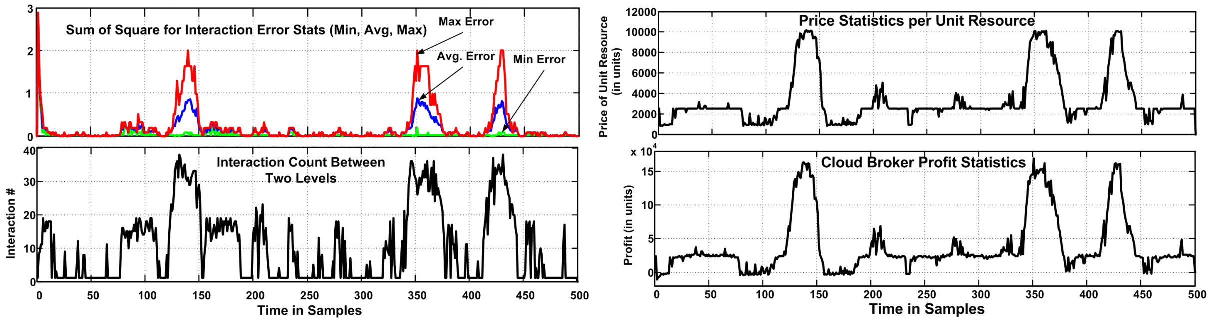
Figure 5 shows the queue size and the response time observed at the service providers by using the proposed interaction balance based resource allocation approach. By comparing the observed queue size among service providers in Figure 5(a), it is obvious that the queue size of the service providers $sp3$ and $sp4$ is lower than the queue size at the service providers $sp1$ and $sp2$, where incoming request rates are in similar range. The primary reason of this phenomenon is the higher penalty on SLAs (see Figure 3(a)) in the cost function of service provider $sp3$ and $sp4$. Similar observation is also valid for the response time statistics in Figure 5(b). This lower queue size and response time on service providers $sp3$ and $sp4$ do not hold true during a few time samples, when the incoming web request



(a) Queue Size Statistics at Each Service Provider.

(b) Response Time Statistics at Each Service Provider.

Figure 5: Queue Size and Response Time Statistics at Each Service Provider.



(a) Interaction Statistics between the Cloud Broker and Service Providers.

(b) Unit Price and Profit Statistics at the Cloud Broker.

Figure 6: Interaction Error and Cloud Broker's Profitability Statistics by Using Equation 6.

rate towards service providers $sp3$ and $sp4$ is much higher than the total towards service providers $sp1$ and $sp2$, such as time samples 0-50, 120-140, etc.

Interaction Count and Interaction Error

Figure 6(a) shows the number of interactions between the service providers and a cloud broker for determining the optimal unit price $\beta(k)$ of the cloud resources and the share of cloud resources $\alpha_i(k)$ assigned to each service provider for maximizing the cloud broker profit and maintaing the SLAs of the service provider. According to this figure, the number of interactions between the service providers and the cloud broker varies with the variation in total incoming web requests towards the cloud broker (see Figure 4). The number of interactions increase from 1 to 40 when the total incoming web requests are either minimum or maximum. In case of a low rate of incoming web requests, not all of the service providers require all the available computational resource, therefore a large number of interactions take place to decrease the unit price of the resource to facilitate the 100% resource allocation. Similarly, when the incoming web request rate is extremely high, large number of interactions are observed due to increasing the unit price of resource, which also reflects that these service providers are willing to pay more for acquiring more resources to maintain their SLAs.

Pricing of the Resource and Profitability

Figure 6(b) shows the unit price of the computational resources during simulation and profitability of the cloud broker, which is calculated using Equation 6. By comparing Figure 6(b) with Figure 4(b), it is evident that the unit price of the computational resource increases when the total incoming web requests are increasing, and it finally results in an increased profit of the cloud broker. This observation can be explained by the basic mechanism of the proposed approach, that in case of limited resources, a cloud broker can increase the price, which will maximize its profit. All the service providers will either compete for the limited resources by paying the increased price or decrease their share to maintain the cost of operation. At a few instances of simulation, when the incoming request rate is too low, profitability of the cloud broker decreases because the unit price of the cloud resource is too low and some of the resources remain unassigned too.

5. Remarks on the Proposed Approach

In this paper, a cloud computing infrastructure is considered for optimally deploying a set of services, on a set of available cloud computing resources by using a cloud broker. According to the simulation results presented in the previous section, the proposed performance management approach derives an optimal resource allocation strategy to maintain the SLAs of the service providers, while at the same time maximizing the profitability of the cloud broker. The proposed approach is developed as a generic framework, which makes it a suitable candidate for being applied to a general class of services and resources. The proposed approach is also adaptive to the variations in the incoming web requests towards the deployed services. This approach is independent of the deployment environment and do not require any prior knowledge of the service's performance behavior with respect to cloud resources at the cloud broker involved in dynamic resource allocation. The proposed approach is distributed because both cloud broker and service providers execute their control algorithms to find optimal values of per unit price and computational resource share. Furthermore, the proposed approach is scalable in the number of service providers as these service providers only interact with the cloud broker. There is no interaction among service providers in the proposed approach as described in the Section 3. Therefore, communication overhead between an existing service provider and cloud provider does not increases by adding another set of service providers to the federated cloud environment. It will only add small overhead of communication in the framework that corresponds to communication between the new service provider and cloud broker.

The detailed performance and the overhead analysis of the proposed interaction balance based approach is already discussed by the authors for a general class of web service deployed in a traditional large scale data center environment [22]. The authors have proposed an interaction balance based performance management approach in data center environment by dynamically changing the load share fraction of incoming web requests among multiple instances of a web service [22]. While in current paper, the proposed interaction balance based approach is applied in a cloud broker based hosting environment by changing the resource allocation among multiple service providers. The authors have also demonstrated that the proposed approach supports dynamic addition of more instances of services or deletion (or failure) of existing instances of services from the data center environment while computing the optimal distribution of resources [22]. This feature exhibits self-configuring and self-healing aspects of autonomic computing property of the proposed approach. Moreover, this distributed control based approach has a lower computational overhead compared to the one of a centralized resource allocation approach [22]. Such computational overhead can be further lowered by tuning the error tolerance ϵ and step length ξ at the coordinator (cloud broker) level control algorithm [22]. At the service provider level, the computational overhead can be further lowered by using more advanced tree search techniques (greedy, pruning, heuristics, and A^*) [24].

6. Conclusion and Future Work

In this paper, a distributed control based performance management approach is introduced for efficiently managing the SLAs of services deployed in a cloud computing environment and for maximizing the profitability of the cloud broker at the same time. In addition, this distributed control based approach provides autonomic computing features to the proposed framework via a feedback based control loop. The proposed approach is adaptive to the rate of the incoming web requests towards services; it dynamically changes the resource allocation such that the service providers can maintain their SLAs. The proposed approach aims to maximize the profit for both the cloud broker and the service providers while computing the optimal resource assignment. Moreover, the performance of the proposed approach can also be manipulated by using various tuning options to reduce the computational overheads and to increase the convergence rate of the distributed control algorithm to calculate the optimal values of the control inputs. To our best knowledge, there is no published research or studies yet about applying the interaction balance method for optimal resource allocation through interactive bidding in cloud computing (or broker) environment.

In the future, the proposed approach will be extended on real cloud computing platforms for performance management of the service providers in a broker-based cloud computing environment, where the cloud resources will also have different reliability index with the varying unit price based on the availability. In this case, the proposed approach will be used to derive an optimal deployment strategy for hosting a set of a wide range of web-based services onto the most suitable set of cloud resources such that, in terms of efficiency and reliability, the overall performance of the system can be optimized, while keeping the profitability considerations intact. In addition, these cloud resources will consist of computing (CPU and memory), storage (disk), and network (bandwidth and redundant connections) resources with varying level of cost and priority in the utility function. The cloud broker will also compute the best service deployment strategy for the set of services on these cloud resources, while maintaining the operation and cost constraints.

Acknowledgments: This work was supported in part by the National Science Foundation (NSF) under grant numbers NSF IIP-1127978 and NSF IIP-1034897 at the NSF Center for Cloud and Autonomic Computing, Mississippi State University, and Knowledge Fund from the Nevada Governor's Office for Economic Development. The statements made herein are solely the responsibility of the authors.

[1] Google apps, `http://www.google.com/apps/intl/en/business/index.html` [Mar 2012].

[2] Amazon elastic compute cloud (amazon ec2), `http://aws.amazon.com/ec2/` [Mar 2012].

[3] Ibm smart cloud, `http://www.ibm.com/cloud-computing/us/en/` [Mar 2012].

[4] Windows azure, `http://www.windowsazure.com` [Mar 2012].

[5] N. Roy, A. Dubey, A. Gokhale, L. Dowdy, A capacity planning process for performance assurance of component-based distributed systems (abstracts only), SIGMETRICS Perform. Eval. Rev. 39 (3) (2011) 16–17.

[6] M. Healey, State of cloud 2011: Time for process maturation, http://reports.informationweek.com/abstract/5/5116/Cloud-Computing/research-2011-state-of-cloud.html [Mar 2012] (Jan 2011).

[7] Cloud broker, http://searchcloudprovider.techtarget.com/definition/cloud-broker [Apr 2014].

[8] S. K. Nair, S. Porwal, T. Dimitrakos, A. J. Ferrer, J. Tordsson, T. Sharif, C. Sheridan, M. Rajarajan, A. U. Khan, Towards secure cloud bursting, brokerage and aggregation, in: 2010 IEEE 8th European Conference on Web Services (ECOWS), IEEE, 2010, pp. 189–196.

[9] R. Mehrotra, S. Srivastava, I. Banicescu, S. Abdelwahed, An interaction balance based approach for autonomic performance management in a cloud computing environment, in: F. Pop, M. Potop-Butucaru (Eds.), Adaptive Resource Management and Scheduling for Cloud Computing, Vol. 8907 of Lecture Notes in Computer Science, Springer International Publishing, 2014, pp. 52–70, http://dx.doi.org/10.1007/978-3-319-13464-2_5.

[10] L. Wu, S. K. Garg, R. Buyya, Sla-based resource allocation for software as a service provider (saas) in cloud computing environments, in: 2011 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), IEEE, 2011, pp. 195–204.

[11] K. Gouda, T. Radhika, M. Akshatha, Priority based resource allocation model for cloud computing, International Journal of Science, Engineering and Technology Research (IJSETR) 2 (1) (2013) 215–219.

[12] C. S. Pawar, R. B. Wagh, Priority based dynamic resource allocation in cloud computing, in: 2012 International Symposium on Cloud and Services Computing (ISCOS), IEEE, 2012, pp. 1–6.

[13] K. Dinesh, G. Poornima, K. Kiruthika, Efficient resources allocation for different jobs in cloud, International Journal of Computer Applications 56 (10) (2012) 30–35.

[14] E. Arianyan, D. Maleki, A. Yari, I. Arianyan, Efficient resource allocation in cloud data centers through genetic algorithm, in: 2012 Sixth International Symposium on Telecommunications (IST), 2012, pp. 566–570.

[15] M. Jebalia, A. B. Letaïfa, M. Hamdi, S. Tabbane, A comparative study on game theoretic approaches for resource allocation in cloud computing architectures, in: 2013 IEEE 22nd International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), IEEE, 2013, pp. 336–341.

[16] S. Zaman, D. Grosu, Combinatorial auction-based allocation of virtual machine instances in clouds, Journal of Parallel and Distributed Computing 73 (4) (2013) 495–508.

[17] O. Rogers, D. Cliff, A financial brokerage model for cloud computing, Journal of Cloud Computing 1 (1) (2012) 1–12.

[18] S. Sundareswaran, A. Squicciarini, D. Lin, A brokerage-based approach for cloud service selection, in: 2012 IEEE 5th International Conference on Cloud Computing (CLOUD), IEEE, 2012, pp. 558–565.

[19] P. Jain, D. Rane, S. Patidar, A novel cloud bursting brokerage and aggregation (cbba) algorithm for multi cloud environment, in: 2012 IEEE Second International Conference on Advanced Computing & Communication Technologies (ACCT), IEEE, 2012, pp. 383–387.

[20] M. G. S. A. Titli;, Systems: Decomposition, Optimisation, and Control, Pergamon Press, 1978.

[21] N. Sadati, A novel approach to coordination of large-scale systems; part ii interaction balance principle, in: IEEE Int'l Conf. on Industrial Technology, 2005, pp. 648 – 654.

[22] R. Mehrotra, S. Abdelwahed, Towards autonomic performance management of large scale data centers using interaction balance principle, Cluster Computing 17 (3) (2014) 979–999.

[23] R. Mehrotra, S. Abdelwahed, A. Erradi, A distributed control approach for autonomic performance management in cloud computing environment, in: 2013 IEEE/ACM 6th International Conference on Utility and Cloud Computing (UCC), 2013, pp. 269–272.

[24] S. Abdelwahed, J. Bai, R. Su, N. Kandasamy, On the application of predictive control techniques for adaptive performance management of computing systems, IEEE Transactions on Network and Service Management 6 (4) (2009) 212–225.

[25] M. Arlitt, T. Jin, Workload characterization of the 1998 world cup web site, Technical Report HPL-99-35R1, Hewlett-Packard Labs (September 1999).

[26] S. A. DeLurgio, Forecasting Principles and Applications, McGraw-Hill, 1998.

[27] R. E. Kalman, A new approach to linear filtering and prediction problems, Transactions of the ASME Journal of Basic Engineering (82 (Series D)) (1960) 35–45, http://www.cs.unc.edu/~welch/kalman/media/pdf/Kalman1960.pdf.

[28] N. Kandasamy, S. Abdelwahed, M. Khandekar, A hierarchical optimization framework for autonomic performance management of distributed computing systems, in: 26th IEEE International Conference on Distributed Computing Systems (ICDCS), 2006, 2006, pp. 9–18.

[29] R. Mehrotra, A. Dubey, S. Abdelwahed, A. Tantawi, A Power-aware Modeling and Autonomic Management Framework for Distributed Computing Systems, Vol. 2, Handbook of Energy-Aware and Green Computing, CRC Press, 2011, Ch. 27, pp. 621 – 648.

Towards an Autonomic Performance Management Approach for Cloud Brokers Using an Interaction Balance Based Methodology

Biographies of Authors:

Rajat Mehrotra

Dr. Rajat Mehrotra is an Innovation Fellow at the Applied Innovation Center for Advanced Analytics (AIC), at Desert Research Institute, Reno, NV. Prior to this, he was Research Associate at Department of Computer Science at University of Virginia, Charlottesville, VA. Dr. Mehrotra finished his PhD in Dec, 2013 at Department of Electrical and Computer Engineering, Mississippi State University. Prior to joining Mississippi State University, he was a senior research engineer at Alcatel-Lucent Development India Pvt. Ltd. in Noida (U.P.), India. Dr. Mehrotra's research includes Autonomic Computing Systems, Cloud Computing Systems, Model Integrated Computing, Cyber Physical Systems, Predictive Modeling, Big Data Systems, and Machine Learning. He has published his research contributions through various research publications and book chapters. He is a member of IEEE, ACM and various honor societies.

Srishti Srivastava

Srishti Srivastava is a Doctoral Candidate at the Department of Computer Science and Engineering at Mississippi State University since August 2010. Her research interests include dynamic load balancing, high performance computing, performance and reliability analysis, optimization, and prediction, and autonomic computing. Presently, she is also a graduate research assistant at the Center for Autonomic Computing at Mississippi State University, which is also one of the four National Science Foundation sites for autonomic computing. Srishti has authored and co-authored a number of articles published in renowned IEEE and ACM conferences. She is the member of the IEEE computer society, ACM, Society for Industrial and Applied Mathematics (SIAM), Computing Research Association (CRA, CRA-W), Anita Borg Institute Grace Hopper Celebration (ABI-GHC), and an honor society of Upsilon Pi Epsilon (UPE).

Ioana Banicescu

Ioana Banicescu is a professor in the Department of Computer Science and Engineering, the Director of the NSF Center for Cloud and Autonomic Computing at Mississippi State University (MSU), and also a Co-Director of the National Science Foundation Center for Cloud and Autonomic Computing. She received the Diploma in Engineering (Electronics and Telecommunications) from Polytechnic University - Bucharest, and the M.S. and the Ph.D. degrees in Computer Science from New York University - Polytechnic Institute. Professor Banicescu's research interests include parallel algorithms, scientific computing, scheduling theory, load balancing algorithms, performance analysis, evaluation, and prediction. Currently, her research focus is on performance optimization for problems in computational science, and autonomic computing. She has given many invited talks at universities, government laboratories, and at various national and international forums in the United States and overseas. She has authored and co-authored more than 100 articles published in journals, books, and conference proceedings.

Professor Banicescu is the recipient of a number of awards for research and scholarship from the National Science Foundation (NSF), including the prestigious NSF CAREER award, three NSF Information Technology awards, and others. She served and continues to serve on numerous research review panels for advanced research grants in the US and Europe, on steering and program committees of a number of international conferences, symposia and workshops, on the Executive Board and Advisory Boards of the IEEE Technical Committee on Parallel Processing (TCPP). She is an Associate Editor of the Cluster Computing journal and International Journal on Computational Science and Engineering.

Sherif Abdelwahed

Sherif Abdelwahed is an Associate Professor with Electrical and Computer Engineering Department at Mississippi State University. He received his Ph.D. degree in Electrical and Computer Engineering from the University of Toronto, Canada, in 2002. From 2000 to 2001, he was a research scientist at Rockwell Scientific Company. From 2002 to 2007 he worked as a research assistant professor with the Institute for Software Integrated Systems at Vanderbilt University. His main research interests include model-based design and analysis of self-managing computation systems, modeling and analysis of distributed real-time systems, automated verification, fault diagnosis techniques, and model-integrated computing. He has published over 90 publications. He is a senior member of the IEEE and member of Sigma Xi.

***Biographies (Photograph)**

*Biographies (Photograph)

***Biographies (Photograph)**

***Biographies (Photograph)**