# An Interaction Balance Based Approach for Autonomic Performance Management in a Cloud Computing Environment

Rajat Mehrotra[*], Srishti Srivastava[†], Ioana Banicescu[†], and Sherif Abdelwahed[‡]
[*] Department of Computer Science, University of Virginia, Charlottesville, VA
[†] Department of Computer Science and Engineering, Mississippi State University, MS
[‡] Department of Electrical and Computer Engineering, Mississippi State University, MS
Email: rajat@cs.virginia.edu, {srishti@hpc, ioana@cse, sherif@ece}.msstate.edu

**Abstract**

In this paper, a performance management approach is introduced that provides dynamic resource allocation for deploying a general class of services over a federated cloud computing infrastructure. This performance management approach is based on distributed control, and is developed by using an interaction balance methodology which has previously been successfully used in developing management solutions for traditional large scale industrial systems. The proposed management approach is used to derive an optimal resource allocation for hosting a set of services in a cloud infrastructure by considering both, the availability and the demand of the cloud computing resources. The primary goals of this performance management approach are to maintain the service level agreements, maximize the profit, and minimize the operating cost for both, the service providers and the cloud brokers. The cloud brokers are considered third party organizations, which work as intermediaries between the service providers and the cloud providers to sublet the rented cloud resources for a fixed time duration and deploy respective services on these resources at a profitable rate. Further, the allocated computing resources are utilized by the service providers to maintain the service level agreements of hosted services by using model-based predictive controllers.

Acknowledgments: This work was supported in part by the National Science Foundation (NSF) under grant number NSF IIP-1127978 and NSF IIP-1034897 at the NSF Center for Cloud and Autonomic Computing, Mississippi State University, and by C-FAR, one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA at the University of Virginia, Charlottesville, VA.

Contact Author: Rajat Mehrotra,
Email: rajat.meh@gmail.com or rajat@cs.virginia.edu
Phone: (571)-294-6583
Address:
Department of Computer Science,
School of Engineering and Applied Science,
University of Virginia,
85 Engineer's Way, P.O. Box 400740
Charlottesville, Virginia 22904-4740

## I. Introduction

There is an increasing trend to host 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 management responsibility to the cloud providers, such as Google Apps [1], Amazon Web Services [2], IBM Cloud [3], and Microsoft Windows Azure [4]. The cloud providers deploy services on multiple computing nodes in accordance with the service level agreements (SLAs) negotiated between the cloud provider and the service provider for a given system resource availability and a specified pay-per-use. According to a survey [5] published in year 2011, 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 their various services) require complex configurations of computing, network, and storage resources. Currently, the 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. In general, the designers of service deployment schemes consider multi-dimensional and mutually conflicting objectives, which include operating costs, SLAs, configuration constraints, resource utilization, and availability of the resources. If the deployment scheme is too optimistic (with an underestimation of the resource requirement), it may result in excessive scarcity of resources, leading to the 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 [6]. 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.

Recently, the service providers started to use a third party named *Cloud Broker*, for selecting the appropriate cloud provider in the interest of service providers, and deploy their services in the selected 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 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 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 is ongoing for its development.

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]. 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 service provider(s) try to maximize their respective profits, such that the 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 incoming http request rate towards the 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 its SLAs 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. The proposed approach is a proof of concept using an interaction-balanced technique for solving the negotiation problem between the cloud broker and the service providers, and represents a step towards a model-based autonomous deployment of robust, adaptive, and reliable services in a cloud computing environment.

This paper is organized as follows. Research efforts made by other groups are presented in Section II, and the proposed performance management approach is introduced in Section III. The application of the proposed approach in a typical cloud infrastructure is demonstrated in Section IV, and the benefits of the approach are highlighted in Section V. Finally, conclusions and future work are presented in Section VI.

## II. Related Work

### A. *Resource Allocation in 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 [9]. 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 [10], [11]. 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 [12], 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 [13].

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 [14]. 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, and others. 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 [15]. This approach uses three different schemes: 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.

### B. *Significance of Cloud Brokers*

The approaches discussed in the previous subsection present solutions to address the challenges of efficient resource management between a cloud provider and a (or many) 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 the 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], [16]. The cloud brokers are the most promising solution available to deal with the complexities of a federated cloud environment. Therefore, there is a tremendous need for applying efficient resource allocation and management in a cloud broker model.

Recently, researchers in this domain have proposed various solutions to address the problem at different levels, such as, managing the information of a large number of cloud service providers via a unique indexing technique [17], enabling the cloud-computing services broker to use derivative contracts in combination to reliably providing cheaper resources to the consumer and predicting future usage [16], proposing novel algorithms for a secure cloud bursting and aggregation operation, using a secure sharing mechanism such that the cloud resources are shared in a secure manner among different cloud environments [18], and others. 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 service providers' 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.

### *C. Large Scale Control-Based Performance Management Approaches*

Large scale control-based methods have recently emerged as 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) [19]. 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" [19].

Furthermore, application performance issues in cloud broker environment can be addressed by service providers through developing service specific controllers that can manage the web service requirements 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 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 this paper:** In this paper, a distributed control-based performance management approach is developed for performance management of services hosted in distributed cloud broker environment. The proposed algorithm utilizes the interaction balance based management 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 computing infrastructure. In this performance 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 [19], [20], [21], [22] 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. 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 provider) profit maximization in cloud computing systems. Therefore, our work is a novel contribution to this problem domain.

## III. 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 price $\beta_{ini}$ of the unit 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 resource price per unit, 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 $\epsilon$ left unassigned), or the unit price becomes lower than the minimum (or threshold) price $\beta_{min}$, or the service providers refuse to pay the current price considered as $\beta_{max}$. 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 $k$.

In this scenario, $N$ services need to be deployed in the cloud infrastructure through a cloud broker as shown in Figure 1. 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 provider. 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 $i$ can be described by the following set of equations.

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

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

$$u_i(k) = \alpha_i(k)\,U(k) \tag{3}$$

where $[a]^+ = \max(0, a)$, $q_i(k)$ is the local queue size of the service $i$, and $\hat{\omega}_i(k)$ is the expected arrival rate of http 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 (in frequency) acquired by the $i$-th service provider 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).

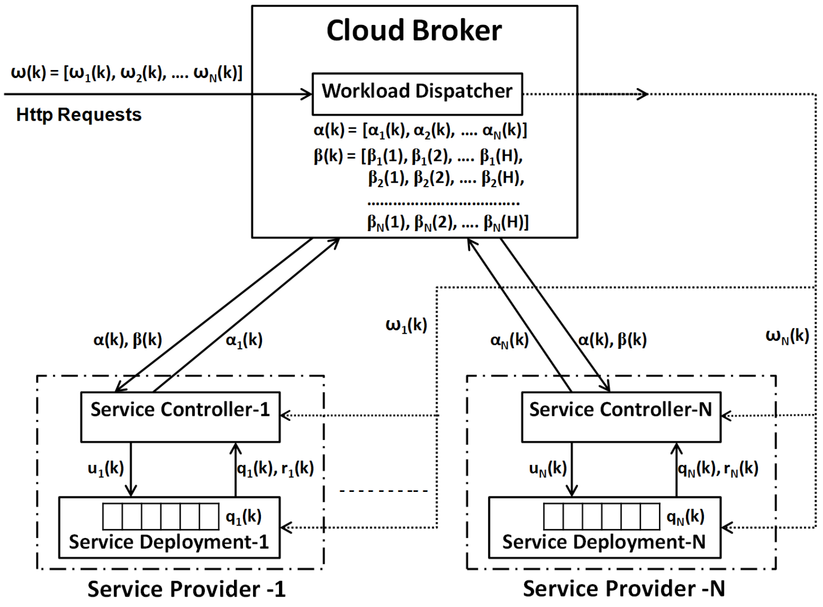


Fig. 1. A Two-Level Distributed Control Structure for Resource Allocation and Performance Management

These equations (1, 2, and 3) represent the aggregate performance behavior of the service providers with respect to 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], [21]. 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 control problems.

The operating cost of the service provider $i$ 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. 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 service $i$ affects the cost function $J_i(k)$ as well as $J_j(k)$, where $j \neq i$.

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

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

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

Here, $k = 1, 2, .., H$ represents the sampling instances in the trajectory of the system operation. 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 providers. 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 $\beta_{ini}$) of the cloud resources too in order to encourage the service providers for extra resource allocation. Otherwise, the cloud broker can also increase the price of a resource unit (from the initial unit price $\beta_{ini}$) when there are not enough resources available to satisfy the requirement of all the service providers. The total revenue of the cloud provider 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.*

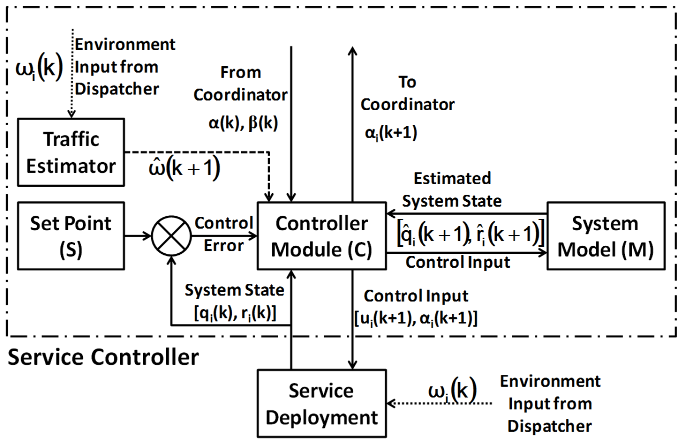


Fig. 2. Service Provider Level Control Structure

## A. Problem Decomposition

The cloud broker considered here hosts $N$ service providers, which are coupled only through the cloud resources $u_i(k)$ and cloud level cost function $J(k)$. The cloud 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 $\alpha_j^*$ 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 $\sum_{j=1, j\neq i}^{N} \alpha_j^*(k) + \alpha_i(k) = 1$. The Lagrangian of each service provider $L_i$ 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 $\beta_i \in R^H$ is the price vector corresponding to the service provider $i$ that is extracted from the Lagrange multiplier vector $\beta$ received from the cloud broker (see Figure 1), where $\beta \in R^{NH}$. The price vector $\beta$ 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 provider. If service providers miss the constraint negatively $(1 < \sum_{i=1}^{N} \alpha_i^*(k))$, the Lagrange multiplier $\beta$ 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 $\beta$ will decrease, which is the same as decreasing the price of the resources if the demand is lower than the availability to encourage maximum allocation.

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 $\beta$, so that the interaction error (see Equation 9) can become less than a pre-defined small value $\epsilon$, where $\epsilon$ can be chosen in percentage of the resource not assigned to any of the service provides (5%). It indicates that cloud broker will keep running the auction until at least 95% (or maximum 105%) 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 $\beta(k)$.

**Algorithm 1** 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 $i$, $\alpha_i^l = [\alpha_i^l(1), \alpha_i^l(2), .., \alpha_i^l(H)]$
 1: service provider state $x_i(k) = [q_i(k)\ r_i(k)]$
 2: $s_k := x(k)$, $Cost(k) = 0$
 3: for all fraction set $f \in F_i$ do
 4:    $\alpha_i^l(1) = \alpha_i^l(2) = .. = \alpha_i^l(H) = f$ /* Same resource share at each step */
 5:    for all $k$ within prediction horizon of depth $H$ do
 6:       $s_{k+1} := \phi$
 7:       for all $x \in s_k$ do
 8:          Compute $\hat{q}_i(k+1)$ /* using Equation 1 */
 9:          Compute $\hat{r}_i(k+1)$ /* using Equation 2 */
10:          Compute $L_i(k+1)$ /* using Equation 7 */
11:          $Cost(k+1) := Cost(k) + L_i(k+1)$
12:          $\hat{x} = [\hat{q}_i(k+1)\ \hat{r}_i(k+1)]$
13:          $s_{k+1} := s_{k+1} \cup \{\hat{x}\}$
14:       end for
15:       $k := k+1$
16:    end for
17: end for
18: Find $x_{min} \in s_N$ having minimum $Cost(k)$
19: Choose $f$ with minimum $L_i(k)$, where $f = \alpha_i^{*(l)} = [\alpha_i^{*(l)}(k), .., \alpha_i^{*(l)}(k+H-1)]$
20: return $\alpha_i^{*(l)}$
```

## B. 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 for the resource fraction $\alpha_i$, and with that determine the optimal value of the control inputs $u_i(k)$, which 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 1, which minimizes the Lagrangian $L_i(k)$ in Equation 7 through a tree search method [23]. Here, $l$ indicates the iteration instance between the service provider and the coordinator within time sample $k$.
2) Forward the optimal values of $\alpha_i^{*(l)}$ to the cloud broker.

## C. Cloud Broker Level Control

At the cloud broker level, the goal is to update the values of the Lagrange multipliers $\beta$ 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 [19] as per following set of equations:

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

Where, $\xi^l$ 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) $\ell_2$-norm. The main steps of the algorithm at the cloud broker level are as follows:

1) Set initial values of the Lagrange multipliers vector $\beta$ as the initial price of the unit resource ($\beta_{ini}$), and forward it to the service provider level controllers.
2) The cloud broker uses the values of $\alpha_i^*$ received from the service provider to calculate the interaction error $e$ using Equation 9.
3) If $\|e^l\|_2 \leq \epsilon$, 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 $\beta$ for the next iteration by using Equation 11, 12, and 13. If calculated value of $\beta$ is more than the maximum unit price ($\beta_{max}$) of the resources, or less than the minimum unit price ($\beta_{min}$), stop and assign the requested amount of cloud resources to each service provider. Otherwise, send this updated value of $\beta$ to the service 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.

### *D. Forecasting the Environmental Inputs at each Service Provider:*

These services are deployed in a dynamic and open distributed environment, where the incoming http requests are generated from external users (or clients) that cannot be controlled by the services. In addition, these http requests show a cyclic pattern in the arrival rate based upon the service popularity and time of the day [24]. These http requests vary significantly within a short duration of a few minutes. However, this variation in the arrival rate of the incoming http requests can be estimated within certain accuracy using Autoregressive Moving Average (ARIMA) filters [25] or Kalman filters [26], as done in earlier work [27], [28]. In this paper, an ARIMA filter based prediction module is developed to estimate the future environmental input at each of the service providers $\hat{\omega}_i(k)$ (http requests) in equation 14, which is similar to the approach used in [27]. This module is shown in Figure 2.

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

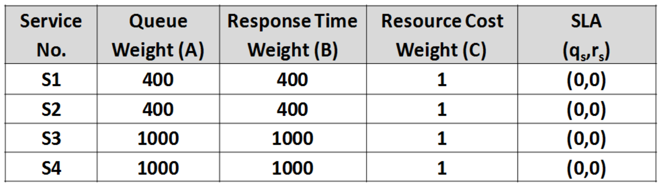
where, $\gamma_{i1}$ and $\gamma_{i2}$ are user specified weights on the current and previous arrival rates. $\bar{\omega}(k-3, r)$ represents the average value of the environment inputs between time samples $k-3$ and $k-3-r$. This prediction module continuously monitors the arrival rate of the incoming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

## IV. 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 manage the performance of four service providers that are hosting their respective service in a cloud computing environment. Details of the simulations are described in the following subsections.

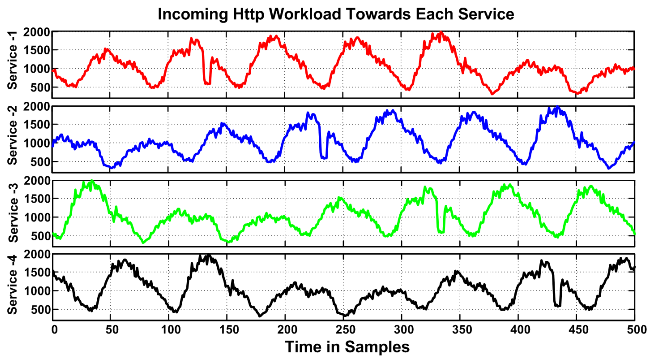
### *A. Service Provider Level Control Setup*

The service provider level controller dynamics is shown in Figure 2. Each of the service providers receives incoming http requests $\omega_i(k)$ from the Dispatcher at the cloud broker during time sample $k$. Each service deployment processes the incoming http 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 $\beta$ 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 1. The calculated resource share $\alpha_i(k+1)$ is sent back to the cloud broker to calculate the interaction error $e$.



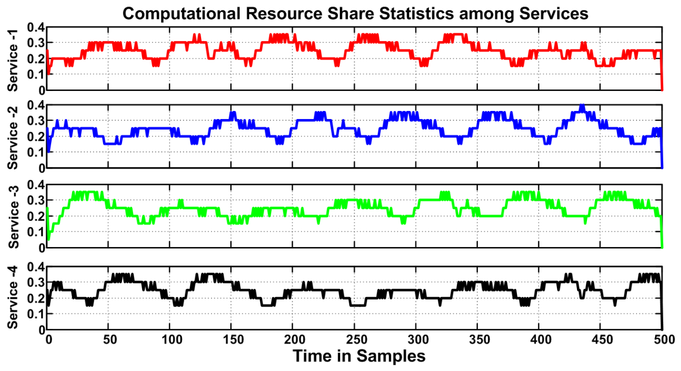
| Service No. | Queue Weight (A) | Response Time Weight (B) | Resource Cost Weight (C) | SLA $(q_s,r_s)$ |
|---|---|---|---|---|
| S1 | 400 | 400 | 1 | (0,0) |
| S2 | 400 | 400 | 1 | (0,0) |
| S3 | 1000 | 1000 | 1 | (0,0) |
| S4 | 1000 | 1000 | 1 | (0,0) |

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

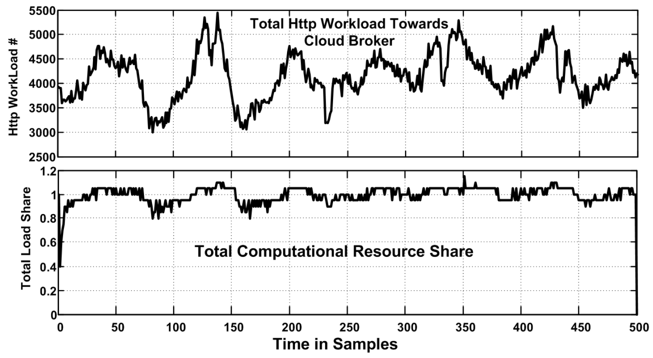


(b) Simulated Incoming http Workload Towards Services.

Fig. 3. Simulation Parameters and Generated http Workload for each Service Provider.



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



(b) Total Http Workload and Total Resource Share Statistics.

Fig. 4. Computational Resource Sharing Among Service Providers and Total Workload Analysis for the Cloud Broker.

## *B. Simulation Setup*

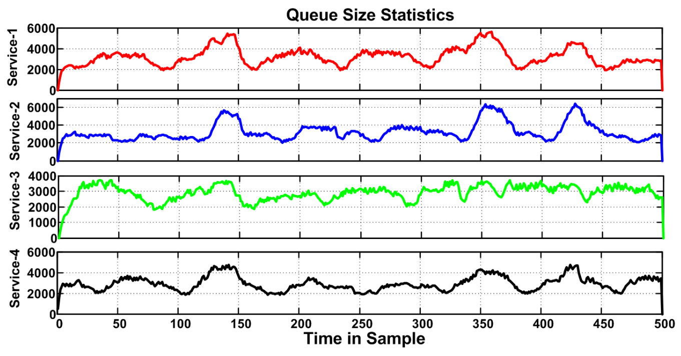
The simulation settings and the coefficients used in the cost function are shown in Figure 3(a). The Service provider 3 and 4 are assigned higher penalty for queue size and response time compared to the service provider 1 and 2. Therefore, the service provider 3 and 4 are expected to choose higher amount of computational resource compared to service provider 1 and 2. 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 http workloads (see Figure 3(b)) are generated for each services by utilizing the 1998 Football World Cup [24] traces. The tolerance value $\epsilon$ of the interaction error is set to 0.05 and the lookahead horizon $H$ is set to 2. In addition, the minimum ($\beta_{min}$), maximum ($\beta_{max}$), and initial ($\beta_{ini}$) unit price of cloud computing resource is set to 1000, 10000, and 2500 pricing unit, respectively. The total available cloud resources 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 http workload for each service.

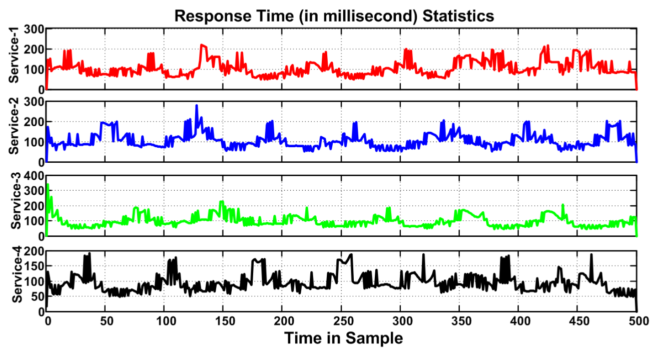
## *C. Simulation Results*

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, extremely dynamic workloads (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 http 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 extremely low rate of incoming http requests towards the services. The results of this simulation are shown in Figure 4, 5, and 6.

***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 http requests (see Figure 3(b)). Initially, service provider 3 receives a higher

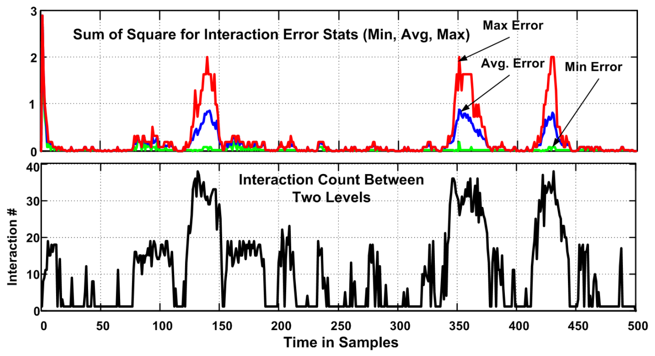


(a) Queue Size Statistics at each Service Provider.

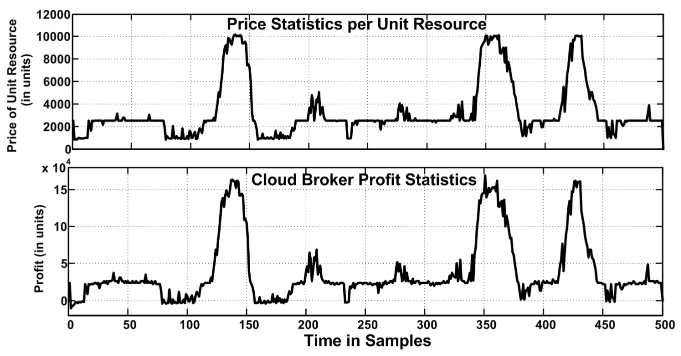


(b) Response Time Statistics at each Service Provider.

Fig. 5. Queue Size and Response Time Statistics at each Service Provider.



(a) Interactions between Cloud Broker and Service Providers.



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

Fig. 6. Interaction Error and Cloud Broker's Profitability Statistics by using Equation 6.

number of http requests compared to other service providers, therefore in the beginning, service provider 3 acquires maximum share of computational resources. Similar phenomenon is observed in case of service provider 1, 2, and 4 during other time samples. Furthermore, the resource allocations adapts to the changes in the workload 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 http workload (sum of the workload 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 http workload is 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 http 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.

***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 3 and 4 is lower than the queue size at the service providers 1 and 2, 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 3 and 4. Similar observation is also valid for the response time statistics in Figure 5(b). This lower queue size and response time on service providers 3 and 4 do not hold true during a few time samples, when the incoming http request rate towards service provider 3 and 4 is much higher than the total towards service providers 1 and 2, 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 http workload towards the cloud broker (see Figure 4). The number of interactions increase from $1$ to $40$ when the total incoming http requests are either minimum or maximum. In case of a low rate of http 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 http request rate is extremely high, these interactions take place for increasing the price of a unit 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 http workload is 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 decreases because price per unit of the cloud resource is too low and some of the resources remain unassigned too.

## V. Benefits of 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 http workload towards the hosted services. This approach is independent of the deployment environment and do not require any prior knowledge of the service provider performance behavior with respect to cloud resources at the cloud broker involved in dynamic resource allocation. Furthermore, the proposed approach is scalable in the number of service providers as these service providers only interact with the cloud broker.

The detailed performance and the overhead analysis of the proposed approach is already done by the authors [21]. In addition, the authors have demonstrated that the proposed approach supports dynamic addition of more services or deletion of existing services from the distributed environment while computing the optimal distribution of resources [21]. Moreover, this distributed control based approach has a lower computational overhead compared to the one of a centralized resource allocation approach that uses a large number of service providers [21]. In addition, this existing computational overhead can be further lowered by tuning the error tolerance $\epsilon$ and step length $\xi$ at the cloud broker level control algorithm. At the service provider level, the computational overhead can be lowered by using more advanced tree search techniques (greedy, pruning, heuristics, and $A^*$) [23].

## VI. Conclusion and Future Work

In this paper, a distributed control-based performance management approach is introduced for efficiently managing the SLAs of a service deployed in a cloud computing environment and for maximizing the profitability of the cloud broker at the same time. The proposed approach is adaptive to the rate of the incoming http requests towards services and dynamically changes the resource allocation such that the service providers can maintain their SLAs. The proposed approach is novel in the terms of giving priority to the profit maximization for both the cloud broker and service providers while computing the optimal resource assignment. None of the research groups in academia or industry have published research or studies 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 per unit price. The proposed approach will be used to derive an optimal deployment strategy for hosting a set of a wide range of services onto the most suitable set of cloud resources such that the overall performance of the system, in terms of efficiency and reliability, is optimized while keeping the profitability considerations intact.

## References

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

[6] Nilabja Roy, Abhishek Dubey, Aniruddha Gokhale, and Larry Dowdy. A capacity planning process for performance assurance of component-based distributed systems (abstracts only). *SIGMETRICS Perform. Eval. Rev.*, 39(3):16–17, December 2011.

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

[8] Srijith K Nair, Sakshi Porwal, Theo Dimitrakos, Ana Juan Ferrer, Johan Tordsson, Tabassum Sharif, Craig Sheridan, Muttukrishnan Rajarajan, and Afnan Ullah Khan. Towards secure cloud bursting, brokerage and aggregation. In *Web Services (ECOWS), 2010 IEEE 8th European Conference on*, pages 189–196. IEEE, 2010.

[9] Linlin Wu, Saurabh Kumar Garg, and Rajkumar Buyya. Sla-based resource allocation for software as a service provider (saas) in cloud computing environments. In *Cluster, Cloud and Grid Computing (CCGrid), 2011 11th IEEE/ACM International Symposium on*, pages 195–204. IEEE, 2011.

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

[11] Chandrashekhar S Pawar and Rajnikant B Wagh. Priority based dynamic resource allocation in cloud computing. In *Cloud and Services Computing (ISCOS), 2012 International Symposium on*, pages 1–6. IEEE, 2012.

[12] K Dinesh, G Poornima, and K Kiruthika. Efficient resources allocation for different jobs in cloud. *International Journal of Computer Applications*, 56, 2012.

[13] E. Arianyan, D. Maleki, A. Yari, and I. Arianyan. Efficient resource allocation in cloud data centers through genetic algorithm. In *Telecommunications (IST), 2012 Sixth International Symposium on*, pages 566–570, Nov 2012.

[14] Maha Jebalia, Asma Ben Letaïfa, Mohamed Hamdi, and Sami Tabbane. A comparative study on game theoretic approaches for resource allocation in cloud computing architectures. In *Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2013 IEEE 22nd International Workshop on*, pages 336–341. IEEE, 2013.

[15] Sharrukh Zaman and Daniel Grosu. Combinatorial auction-based allocation of virtual machine instances in clouds. *Journal of Parallel and Distributed Computing*, 73(4):495–508, 2013.

[16] Owen Rogers and Dave Cliff. A financial brokerage model for cloud computing. *Journal of Cloud Computing*, 1(1):1–12, 2012.

[17] Smitha Sundareswaran, Anna Squicciarini, and Dan Lin. A brokerage-based approach for cloud service selection. In *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*, pages 558–565. IEEE, 2012.

[18] Pritesh Jain, Dheeraj Rane, and Shyam Patidar. A novel cloud bursting brokerage and aggregation (cbba) algorithm for multi cloud environment. In *Advanced Computing & Communication Technologies (ACCT), 2012 Second International Conference on*, pages 383–387. IEEE, 2012.

[19] Madan G. Singh; Andre Titli;. *Systems: Decomposition, Optimisation, and Control*. Pergamon Press, 1978.

[20] N. Sadati. A novel approach to coordination of large-scale systems; part ii interaction balance principle. In *IEEE Int'l Conf. on Industrial Technology*, pages 648 – 654, dec. 2005.

[21] Rajat Mehrotra and Sherif Abdelwahed. Towards autonomic performance management of large scale data centers using interaction balance principle. *Cluster Computing*, pages 1–21, 2014.

[22] Rajat Mehrotra, Sherif Abdelwahed, and Abdelkarim Erradi. A distributed control approach for autonomic performance management in cloud computing environment. In *Proceedings of the 2013 IEEE/ACM 6th International Conference on Utility and Cloud Computing*, UCC '13, pages 269–272, Washington, DC, USA, 2013. IEEE Computer Society.

[23] S. Abdelwahed, Jia Bai, Rong Su, and Nagarajan Kandasamy. On the application of predictive control techniques for adaptive performance management of computing systems. *Network and Service Management, IEEE Transactions on*, 6(4):212–225, 2009.

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

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

[26] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME Journal of Basic Engineering*, (82 (Series D)):35–45, 1960.

[27] N. Kandasamy, S. Abdelwahed, and M. Khandekar. A hierarchical optimization framework for autonomic performance management of distributed computing systems. In *Proc. 26th IEEE Int'l Conf. Distributed Computing Systems (ICDCS)*, 2006.

[28] Rajat Mehrotra, Abhishek Dubey, Sherif Abdelwahed, and Asser Tantawi. *A Power-aware Modeling and Autonomic Management Framework for Distributed Computing Systems*. CRC Press, 2011.