A SURVEY OF QUALITY OF SERVICE IN MULTI-TIER WEB APPLICATIONS

Mohamed Ghetas[1], Chan Huah Yong[1] and Putra Sumari[1]
[1] computer science, Universiti Sains Malaysia USM, Malaysia
11800, Pulau Pinang- Malaysia
[mohghattas@gmail.com]
[hychan@usm.my]
[putras@usm.my]
*Corresponding author: Mohamed Ghetas

Received May 26, 2015; accepted October 29, 2015; published January 31, 2016

Abstract

Modern web services have been broadly deployed on the Internet. Most of these services use multi-tier architecture for flexible scaling and software reusability. However, managing the performance of multi-tier web services under dynamic and unpredictable workload, and different resource demands in each tier is a critical problem for a service provider. When offering quality of service assurance with least resource usage costs, web service providers should adopt self-adaptive resource provisioning in each tier. Recently, a number of rule- and model-based approaches have been designed for dynamic resource management in virtualized data centers. This survey investigates the challenges of resource provisioning and provides a competing assessment on the existing approaches. After the evaluation of their benefits and drawbacks, the new research direction to improve the efficiency of resource management and recommendations are introduced.

Keywords: Quality of Service, Cloud Computing, Resource Management, Multi-tier Application

This research was supported by USM short term grant 304/PKOMP/6313168.

http://dx.doi.org/10.3837/tiis.2016.01.014 **ISSN : 1976-7277**

1. Introduction

Virtualization technology has revolutionized the establishment of large-scale data centers all worldwide. These data centers provide on-demand powerful computing resources to various business applications, such as e-commerce, relationship management, and payroll. The underlying architecture providing the service of these applications is generally called web services. Specifically, web service describes the business functionality exposed by online merchants for shopping buyers on the Internet. Online merchants intend to increase their revenue and return of investment by the continuous and consistent accessibility of their business service.

This survey evaluates the different approaches to, existing works on, and open problems of resource management in multi-tier web applications. Section 2 discusses the architecture of multi-tier web applications. The challenges and opportunities of resource management are presented in Section 3. The problem definition of QoS is provided in Section 4. Section 5 investigates the state-of-the-art approaches to providing QoS guarantee. The open problems and recommendation are highlighted in Section 6, the conclusion is introduced in Section 7.

2. Multi-tier Web Application Architecture

The infrastructure of modern web applications employs multi-tier architecture to offer flexibility, a modular approach, scalability, and reliability in deploying web services [1]. **Fig. 1** shows an example of an e-commerce application that consists of three tiers: the web server, application, and database tiers.

In three-tiered architecture, the front-end web server acts as a presentation layer, which utilizes one or more worker threads to process incoming http requests. Whenever a new request is received, the web server assigns the request to a free worker. Each worker is capable of processing a single request before receiving another one. After the processing of the request, the worker can either send a response to the client in case of a static content or forwards the request to the second tier in case of a complex and dynamic content, then it moves to the block state. The blocked worker is woken up once the web server receives a response from the second tier, and then it sends the response to the client. Therefore, the web tier has three functions: (1) accepting/rejecting incoming requests and serving static content, (2) forwarding complex requests to the application server, and (3) receiving the response from the application server and sending a reply back to the client. Microsoft Internet Information Server (IIS) and Apache are good examples of web servers [2].

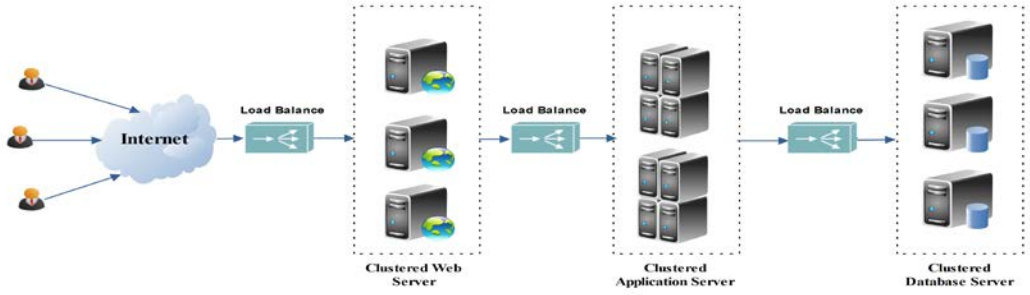


Fig. 1. Architecture of three tier E-Commerce application

The application server implements a complex business logic and resides between the web server and database tiers. It provides functionalities for security, session state, and database access. Application server is product-based and commonly comprises a servlet engine container that executes Java Servlet, which is considered the basic block of web applications, and an Enterprise JavaBeans (EJB) container that handles transactions, threads, and connection pools. For example, Tomcat [3] is a servlet engine container that implements Java Server Pages (JSPs) and Java Servlet [4]. The functionality of Tomcat is similar to Apache, which holds several worker threads to manage received requests from Apache and forwards these requests to the database tier.

The database tier is considered the data house, which is used to store user accounts, customer orders, and site information. The database engine in a multi-tier application includes Microsoft SQL, Sybase, MySQL, Oracle, and PostgreSQL. The database server uses multithreaded architecture, like web server and application server, but it uses the thread cache in which cache threads (group of threads) can be reused by subsequent SQL queries.

Principally, each application tier is ideally distributed across distinct servers. Furthermore, a tier might be clustered based on the required capacity to sustain the application performance metrics that are represented as SLAs. For example, the Apache server can be run on multiple virtual machines hosted on different nodes, and the number of replicas should be identified dynamically to provide a sufficient capacity.

3. Challenges and Opportunities in Resource Management

Adaptive resource provisioning in multi-tier services is not a trivial task because of the complex multi-tier behavior and changing workload. This section investigates the resource management opportunities and challenges in multi-tier services:

1. The running service requires various types of resources, and every resource has a distinct influence on performance quality. Furthermore, the relationship between the application performance and the required resources is complex and nonlinear [5], [6]. Therefore, designing a performance model is a challenging task.
2. Compared to that in a single tier application, resource provisioning in a multi-tier application is much more difficult because of the interaction between tiers, where each tier requires different on demand resources and has different influences on the QoS.
3. The system's coarse-grained resource management and approximation of nonlinearity to a linear relationship lead to low resource utilization and even performance degradation. For instance, historical-based performance control approaches help in resource provisioning but do not provide the desired QoS [7], [8], [9].
4. Clout providers do not offer QoS assurance on application level performance, such as response time and throughput.

4. Problem Definition of Dynamic Resource Management

The key resource management objective in a multi-tier web application is to allocate the optimal amount of on demand resources with minimum resource costs. Therefore, resource management does not simply provide an adaptive resource provisioning for QoS guarantee but also improves resource utilization. The request processing model is depicted in **Fig. 2**, where

the resource management in multi-tier applications includes admission control, application resource management (vertical and horizontal scaling), and service differentiation.

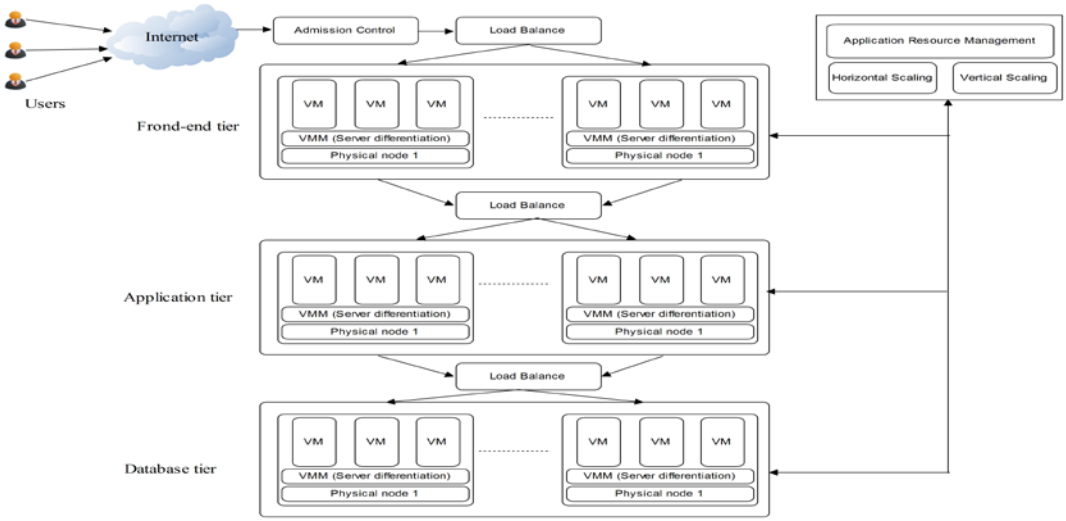


Fig. 2. Resource management in multi-tier application

Resource management algorithms split running time into epochs. The length of every epoch is adjusted to balance the gain and overhead of the algorithm. Considering the fluctuation and unpredictable behavior of the multi-tier application's workload, providing QoS assurance is a challenge. Therefore, admission control helps avoid system overload caused by workload burst and provide QoS guarantee at the beginning of each epoch. It identifies the number of requests that can be served under the current resource allocation and rejects excess requests. However, admission control design is a challenge because the resources required by each request are different and achieving a trade-off between dropped and admitted requests is necessary.

Autonomic resource provisioning in multi-tier applications is essential because of the change in resource demand along with workload fluctuation. Self-adapting resource provisioning not only minimizes the resource costs in terms of energy consumption and operational cost, but also maximizes the revenue of the providers as it keeps the customers satisfied by providing server level assurance.

Resource management has two types of approaches. The first type is known as vertical scaling (scales up and down) in which the resources are allocated to or de-allocated from existing virtual machine (VM) instances at run-time. The application resource management algorithm determines the resource demands of every VM and then the resource scheduler, which resides on each physical node, allocates the estimated resources to the VMs based on their resource demands. If the total on demand resources exceeds the node capacity, the arbitrator controls the resource allocation to the hosted applications to meet the differentiation goal for the end-to-end performance metrics. Rackspace is an example of the running resizing of VMs [10].

The second type of resource provisioning is horizontal scaling in which resource management algorithm determines the number of VM instances based on the workload variation. For instance, AWS EC2 offers dozens of VM instances with different computing capabilities. Horizontal and vertical scaling are complementary to each other and can be used for efficient resource provisioning.

5. The State of Art in Quality of Service Guarantee of Multi-tier Application

Numerous studies have already been performed on QoS in multi-tier applications. In particular, these studies aim to satisfy the performance metrics according to the SLAs between the cloud provider and the cloud customer. Specifically, the literature can be approximately divided into two main types: rule- and model-based approaches. The comparison is detailed in **Table 1**, where the control theoretic-based approaches can be identified as the most suited for providing QoS guarantee in multi-tier systems.

The rule-based approaches based on action selection include fuzzy control logic, reinforcement, and static machine learning. They are capable of identifying on demand resources through the learning system behavior from historical data. The advantage of the rule-based approach is that it provides a model, but it cannot provide QoS guarantee.

In contrast, model-based approaches not only provide QoS assurance but also provide theoretical system performance model and guidelines for analysis, and design and evaluation of the computing system. Theoretical control techniques have been applied to non-linear computing systems for performance assurance and service differentiation by providing a linear approximation of the system dynamics. However, model-dependent approaches may experience inaccuracy because of the workload deviation from those used to identify the system parameters.

Table 1. Applications in each class

	Rule based	**Model based**
Techniques	1. Fuzzy control logic [11], [12], [13], [14], [15], [16], [17], [18], [19]. 2. Reinforcement learning [20], [21], [22]. 3. Statistical machine learning [23], [24].	1. Queueing network [25], [31], [32], [33]. 2. Control theory [25], [26], [27], [28], [29], [30].
Advantages	1. Do not require explicit performance model. 2. Approximate on demand resources from historical data.	1. Offer QoS guarantee. 2. Provide an explicit performance model. 3. Provide rigorous and guidelines for analysis, design and evaluate the system.
Disadvantages	1. Do not provide QoS guarantee.	1. Queuing–based approaches are mean oriented. 2. Control theory may suffer from inaccuracy of modeling dynamic workload.

Queueing network has been introduced into dependent performance model, resource provisioning, and service differentiation. However, most queuing-based approaches are mean-oriented, which means that they base on average response time to evaluate the performance. However, 95th percentile of response time can capture the linearity of the system. Therefore, computing the 95th percentile from the response time distribution of queue-based approaches is time-consuming.

The taxonomy of the characteristic adopted to classify the existing works is depicted in **Fig. 3**. Precisely, the proposed approaches are classified in relation to the following features:

1. *Adopted approach*: Identify the approach for QoS guarantee, whether the algorithm is rule- or model- based.
2. *Resource management algorithm*: Whether the control algorithm uses admission control, service differentiation, or resource scaling to provide QoS assurance.
3. *Goal*: Whether the objective is to provide QoS guarantee or maximize revenue.
4. *Control level*: Whether the control is applied to the node or application level.
5. *Evaluation*: Algorithm is evaluated through test-bed or simulation.
6. *Benchmark*: The type of workload used in algorithm evaluation.

The category of the investigated research concerning the most significant features is demonstrated in **Table 2**. The next subsections focus on resource management techniques that include the methods and strategies for dynamic adapting resource provisioning in multi-tier web applications to obtain the desired QoS.

5.1. Power and Performance Management via Lookahead Control

In [37] the authors addressed the problem of dynamic resource provisioning for a multi-tier application hosted in a virtualized environment. The main objective is to maximize the cloud provider's revenue by reducing the energy consumption, switching the costs and SLA violations. The authors defined resource provisioning as a sequential optimization and applied the limited lookahead control (LLC) to find a near optimal solution.

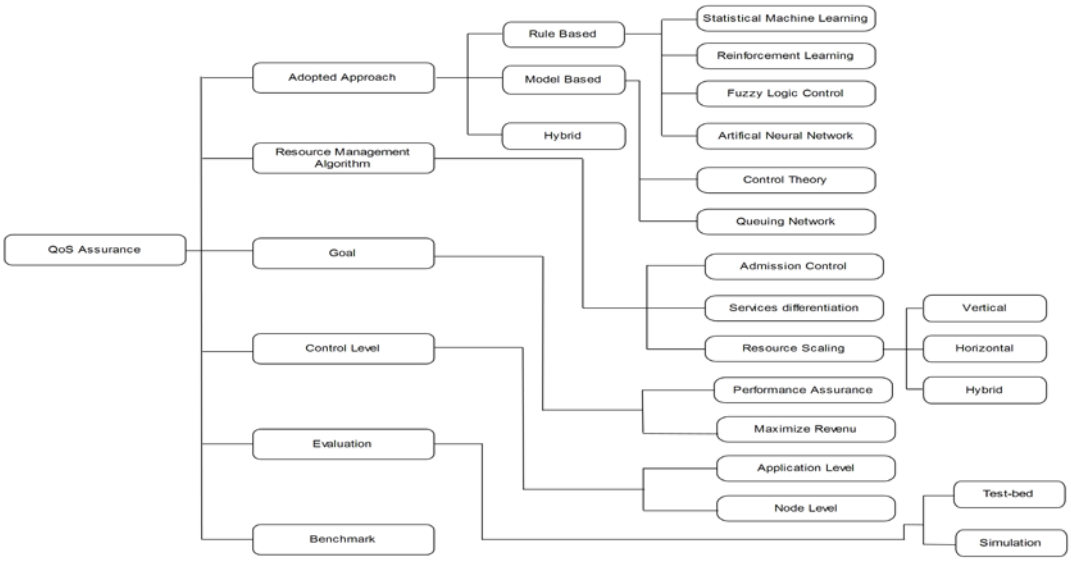


Fig. 3. QoS Management taxonomy

Online optimization controller entails Kalman filter to identify the new system configuration based on the predicted workload. The new system configuration is defined as the number of active servers, number of VMs needed to host the online service, CPU shares to be allocated to each VM, and number of servers to be turned off. Given the uncertainty of the optimization model, the authors provided a risk-aware utility function to account for the risk of excessive switching off of VMs and cost of transient power consumption. To reduce power consumption, the authors used dynamic workload consolidation via offline migration to reduce the number of active servers and turn off unused or under-utilized servers. Furthermore, the authors argued the effect of DVFS in reducing energy consumption and concluded that the use of DVFS has low-power reduction effect.

The authors conducted experiments, and the results showed that LLC achieves power savings of up to 26% with SLA violations of not more than 1.6% of the total requests over 24 hours. However, the proposed model has some limitation. First, the model is an application-dependent and requires a priori knowledge of the sharing CPU of each VM. In addition, the execution time of the optimization controller is approximately 30 minutes for a small system size, which is not applicable for large-scale systems.

5.2. Enabling Cost-aware and Adaptive Elasticity of Multi-tier Applications

The authors in [38] studied the problem of resource scaling in multi-tier applications. The main objective is to minimize the cost of using the multi-tier service. They proposed an adaptive cost-aware scaling approach to reduce the resource usage costs by detecting the bottleneck tier. The scaling process comprises two phases. The first phase defines all the bottleneck tiers. In the second phase, the bottlenecks tiers are resolved in an iterative manner to prevent the creation of other bottlenecks. The multi-tier applications are defined by two parameters. The first parameter is the number of servers that host the application, whereas the second one is the performance requirements based on the SLA.

The proposed framework is known as the Imperial Smart Scaling engine (iSSe). It acts as middleware between the application owner and the infrastructure provider. It consists of five components, namely, the Infrastructure as a Service (IaaS) user portal, monitoring services, capacity estimation service, repository of servers, and deployment service. The cost-aware scaling algorithm detects the trends of the workload change. If the workload change trends increase the request rate, the cost-aware capacity estimation (CACE-Scaling Up) algorithm adds a new server based on the cost per unit decrease in response time. The tier with the least cost is considered as the bottleneck and the new server is added to that tier. The algorithm repeats the system scale-up until the desired response time is obtained. In contrast, if the workload change trends decrease the requests rate, the CACE-Scale Down algorithm iteratively removes one server each time from the tiers, where removing one server does not violate the SLAs.

The authors evaluated the algorithm throughout several experimental studies on different workload types. The results showed that, the algorithm is capable of identifying the bottleneck tier and performing a system scale-up within 2–3 minutes to restore the response time as the workload volume increases. Similarly, the algorithm performs system scale-down to reduce resource usage costs while maintaining the response time target. Moreover, the results showed that the CACE algorithm outperforms both the pre-defined policies (PBS) and tier-dividing scaling (TDS) in terms of less SLA violation time.

5.3. Optimal Cloud Resource Auto-Scaling for Web Applications

In [39] the authors investigated the problem of VM-level auto scaling for web applications. The objective is to achieve cost-latency trade-off by reducing resource provisioning cost and satisfying the QoS based on SLAs. The authors defined VMs scaling as a cost-latency trade-off optimization problem. In each time unit, the scaling algorithm uses a linear regression model to predict the number of requests based on historical data. The authors reduced the feature space of historical usage by finding the top time fragments that have high correlation with the current time unit.

A theoretical performance model based on the queuing network theory and Marko's chain is presented. The performance model makes a decision for scaling system based on the predicted latency time. Furthermore, the scaling algorithm applies the performance model and multi-objective optimization to find the optimal number of VMs to be allocated or

de-allocated to reduce resource usage costs and meet the performance metrics.

To evaluate the effectiveness of the proposed scheme, the authors conducted experiments and used three data sets: AOL, Sogou, and UTSlib. The results showed that linear regression can predict future web requests with low prediction error. In addition, the authors compared the proposed scheme with PEAK, PEAK (× ¾), and CAP (× ¾). The PEAK approach always allocates VMs for the worst-case web requests rate, whereas PEAK (× ¾) only allocates VMs for ¾ of the web requests peak value. CAP (× ¾) sets the number of VMs to two times the number that can meet the predicted web requests rate. The comparison indicated that the proposed scheme achieves better cost saving and minimum SLA violation.

One of the limitations of the proposed algorithm is that, the performance model is application-dependent because the estimation of service time and prediction padding should be adjusted for each application.

5.4. Feedback Control Resource Management in Multiple Web Applications

In [36] the authors addressed the problem of resource scaling in multi-tier applications in the IaaS cloud. The main objective is to reduce the resource usage costs. The proposed scaling algorithm (ARVUE) based on reactive feedback control uses historical resource utilization to estimate the amount of required on demand resources to meet the QoS. ARVUE deploys multiple web applications on a single VM simultaneously, and a fraction of the VM's resources can be added or removed from hosted applications based on workload intensity. In addition, it uses extra VMs to support sudden workload peaks.

ARVUE consists of global and local controllers. The global controller employs the derivative feedback control theory to determine the number of application server tiers and application instances. It makes a decision based on a predefined lower and upper utilization thresholds to identify the bottleneck tier. In contrast, the local controller manages the process of deploying and un-deploying web applications. Moreover, it collects the resource utilization of VMs and delivers them to the global controller. The authors evaluated the effectiveness of ARVUE through a discrete-event simulation and a prototype. The experimental used a synthetic workload with a varied number of active sessions. The results confirmed that sharing VMs resources among different web applications can drastically minimize the total number of VMs. Moreover, the use of extra VMs to handle sudden workload peaks is an appreciated method.

The limitations of ARVE algorithms are: the lower and upper thresholds' values should be tuned individually based on each application's behavior, the additional VMs that accommodate workload peaks incur excess power consumption, and the algorithm does not account for the migration costs incurred during the movement of applications between two servers.

5.5. Adaptive Resource Provisioning for Read Intensive Multi-tier Applications

The authors in [41] investigated the problem of dynamic resource provisioning for multi-tier web applications. The primary objective is to provide QoS assurance. To achieve this goal, the authors proposed adaptive resource provisioning to automatically detect, resolve the bottleneck tier, and react to over-provisioning resources.

The reactive scaling-up algorithm periodically monitors the resource utilization of virtual machines, reads the proxy server log file, and estimates the response time for static and dynamic requests based on the 95th percentile of the average response time. If the estimated response times are above a predefined threshold, the algorithm identifies the bottleneck tiers and adds a new server to the bottlenecked tier. In contrast, a regression-based predictive model

predicts the required VMs at the beginning of each time interval. The predictive model periodically updates the coefficient parameters every time a new observation is received. On one hand, similar to the reactive model, the predictive model refers to the static and dynamic requests rate to determine the required VMs for the web server tier. On the other hand, it predicts the number of database VMs based on dynamic requests rate only.

In order to evaluate the proposed algorithms and investigate the throughput of the system with static resource allocation under a limited system capacity. The authors conducted test-bed experimental. The experimental showed the effectiveness of the resource over-provisioning algorithm on increasing the system's throughput compared to the static resource allocation. One of the limitations of the proposed scaling algorithms is that they do not provide QoS guarantee because of the unpredictable workload and nonlinear behavior of multi-tier web applications.

5.6 Economical and Robust Provisioning of N-Tier Cloud Workloads

The authors in [33] addressed the problem of resource management in multi-tier applications under resources budget and performance constraints. The main objective is to minimize the resource usage costs and maintain the desired QoS despite of workload fluctuation. The problem of resource provisioning can be roughly divided into two sub-problems: estimation of on demand resources and resources' partitioning.

The multi-tier web application is modelled as a tandem queue. Then, the queuing network theory is entitled to build the performance model of the system. The application level controller employs feedback controller to estimate the amount of required resources to satisfy the end-to-end response time. The feedback controller uses an auto regressive-moving-average (ARMA) model [34] to approximate the nonlinear relationship between the allocated resources and the end-to-end response time. By contrast, the local controller at the container level obtains the optimal resource partition between hosted VMs that minimizes the resource usage costs and end-to-end response time. Consequently, the Lagrange function is employed to obtain the optimal solution for the resource allocation problem.

In order to evaluate the robustness of the proposed algorithm, the authors conducted several experiments with different workload types. The results showed that the proposed algorithm outperforms the utilization and equal shares approaches; it can save resources up to 20%. The limitations of the proposed algorithm are the proposed algorithm needs to solve more complex optimization problems, the prediction model is not capable of providing accurate resource estimation, and the proposed algorithm is centralized, and the performance model cannot capture the shape of the response time distribution.

5.7. Online Self-reconfiguration in Large-scale Data Centers

In [42] the authors investigated the problem of energy efficiency in large-scale data centers hosting multiple applications spanning over multiple virtual machines. The objective is to minimize the number of used servers to minimize consumed energy under the performance metrics constraints. The proposed self-configuration framework includes dispatcher, VM managers (VMMs), and reconfiguration policy generator (RPG). The dispatcher distributes the incoming requests of an application to its VMs. VMM is responsible of VMs migration, deployment, and de-deployment. It applies Brown's quadratic exponential smoothing to predict workload intensity and sends the predicted values to the RPG. The RPG applies a genetic algorithm to find the optimal system configuration and uses a push function to accelerate the process of searching for a new system configuration. Furthermore, the genetic

algorithm uses the push function and energy efficiency to measure the fitness of the generation. The new system configuration is delivered to the VMM for reallocation or de-allocation of VMs.

The experimental results demonstrated that Brown's quadratic exponential smoothing can predict the workload intensity with high accuracy. Moreover, the self-configuration framework outperforms other resource managements, such as TSSP07 [35], and saves energy up to 25%. The limitations of the proposed algorithm are the centralized controller and difficulties in tuning the key parameters, such as population size, mutation rates, and crossover.

5.8. Agile Dynamic Provisioning of Multi-Tier Internet Applications

The authors [40] addressed the problem of dynamic VM provisioning in multi-tier applications that have long- and short-term workload variation. The objective is to determine the amount of resources and when they are allocated to maintain the desired QoS. The proposed resource management framework includes a nucleus software component that resides in every server and periodically measures the performance and resource utilization and delivers these measurements to the control panel. The control panel applies a queuing-based analytical model to estimate the required capacity to be allocated to each tier to meet the performance metrics based on the decomposition and the end-to-end response time across different tiers.

The resource management framework has two distinct modules, namely, predictive and reactive. The predictive module predicts the trends of workload variation based on historical observations, and then it estimates the required resources to accommodate long-term workload changing. By contrast, the reactive module corrects the errors caused by the deviation of the long-term workload or by unanticipated flash crowds. However, allocating new resources is time consuming; therefore, resources are switched from one application to another to reduce VM deployment overhead. The system is ramped down in case of light workload. Two approaches are employed: fixed-rate and measurement-based ramp downs. The resources in the fixed-rate ramp down are switched from under-loaded application to another in a fixed amount of time. By contrast, the measurement-based ramp down is a conservative approach and significantly depends on decreasing the rate of the resource usage of the existing session.

The results from the experiment and simulation showed that the proposed resource management framework accurately identifies the bottleneck tier and precisely determines the required capacity. The comparison with the block box approach showed the superiority of the proposed framework because it accounts for the replication constraints imposed on each tier.

The limitations of the proposed framework are as follows: it does not address the consolidation of data-intensive services, such as database scalability, and does not consider how the system releases unused resources.

5.9. A Regression-Based Analytic Model for Dynamic Resource Provisioning

In [43], the authors studied the problem of capacity planning and dynamic resource provisioning for multi-tier web applications. The objective is to meet the performance requirements despite the time-varying workload. A theoretical framework based on the regression model and queuing theory is applied to evaluate the required resources of a complex transaction.

The regression model approximates the on-demand CPU required by each transaction; the model parameters are updated every time interval based on available monitoring and collected resource utilization. The analytical model represents the multi-tier application as a closed

system through a series of networking queues. The model uses the results from the regression-based model to dynamically determine the capacity of the system under different transaction mixes. The model utilizes the mean-value analysis algorithm to identify the average system throughput and response time.

The results from the experiment demonstrated that the regression-based model can approximate the cost of transactions of the front tiers with higher accuracy than the cost of the transaction at database tiers. By contrast, the large monitoring window has a significant effect on the approximated CPU cost at database tiers, but it has less effect on the accuracy of the CPU cost at web server tiers. The results from the analytical model exactly match those from the experiment for shopping and ordering mixes. The results also indicated the correctness of simplifying the session-based to transaction-based traffic, and their performance is consistent with the experimental results.

The limitations of the proposed resource management algorithm are as follows: it is application dependent and is incapable of capturing a dynamic workload, the prediction model is overestimated over 15% under browsing mixes, the detection of the bottleneck tier is missed, and how resources are added to absorb the workload spike is not provided.

5.10. Multi-Tiered On-Demand Resource Scheduling for VM-Based Data Center

The authors in [44] studied the problem of dynamic resource provisioning for hosted multi-services in a virtualized environment. The objective is to increase the resource utilization to reduce energy consumption. Each service is deployed on several VMs hosted on different physical servers. The resources are flowing among hosted services based on the priorities of services. The proposed resource management algorithm accounts for CPU and RAM utilization in resource provision decisions. The resource management algorithm ensures the performance of critical and high-priority services by degrading the performance of low-priority services and flowing resources to critical high-priority services when resources compete.

The authors proposed a multi-tier resource scheduling scheme to dynamically flow the resources among VMs and to optimize the resource allocation between services. The multi-tier resource scheduling has been implemented at three different levels: the application-level resource scheduler dispatches the incoming requests to service VMs; the local resource scheduler runs inside the individual physical server and allocates the resources to running VMs on its physical server based on the service priority of the VM; and the global level resource scheduler controls the resource flowing between hosted services, pre-instantiates VMs on a set of physical servers, and provides a slice of the total resources allocated to an application to different VMs.

The limitations of the proposed algorithm are as follows: it does not apply the VM migration to adapt to the placement of VM at run time, the priority of the running services should be explicitly defined, and the utility function should be learned.

5.11. Efficient Server Provisioning with End-to-End Response Time Guarantee

The work in [18] addressed three challenge problems that occur in multi-tier web applications, and these problems are dynamic server provision, end-to-end response time guarantee, and server switching delay. The objective is to minimize the total number of servers allocated to multi-tier web applications while satisfying the end-to-end response time.

However, the problem of server provisioning is optimization and model dependence; the fuzzy logic controller is model independent and can be used for resource provisioning. This controller is capable of capturing the shape of the response time curve and identifying the

number of required servers to satisfy the average and bound 90th end-to-end response times. The inputs to the fuzzy controller are the difference between the desired and the measured values of the end-to-end delay and the error change. The controller based on the rule base is used to infer the number of servers to achieve the desired end-to-end response time. The scaling factor controller automatically adjusts the output of the fuzzy controller by factor alpha to increase the accuracy of the controller.

The authors conducted intensive simulations to evaluate the server provision approach with and without the fuzzy controller. The server optimization aims to use a minimum number of servers to satisfy the end-to-end response time rather than the decomposition-based approach in [40] and the balanced decomposition-based approach. Therefore, the integration of the fuzzy controller in the server provision leads to reducing the number of servers and satisfying the desired end-to-end response time. One of the limitations of the proposed resource management is that it cannot provide a QoS guarantee under dynamic workload changing.

5.12. Virtualization-Based Autonomic Resource Management

The authors [47] studied the problem of resource management to provide differentiated service qualities in multi-tier applications. They proposed an adaptive self-management resource framework based on a probabilistic-based analytical performance model to efficiently allocate resources among applications with different priorities.

The resource management framework (VS–RA) consists of the below elements. The self-configuration manager is responsible for preparing and configuring the virtual machine from the backup pool. Self-healing enhances the system stability by reducing the possibility of failure. Self-optimizing employs utility function to identify the resource allocation policies under time-varying workloads and follows the MAPE (monitoring, analyzing, planning, and executing) approach. Self-protecting protects the system from external threats. Resource allocation is defined as an optimization problem with a well-known objective utility function. A performance model based on queuing theory and probability analysis is introduced to solve the optimization problem. Gamma distribution is employed to find the 90th percentile for the end-to-end response time delivered from the queuing-based performance model. Consequently, the number of finished requests below a threshold value can be found. Queuing theory is applied again to abstract the system as an M/M/1 queue to reduce the performance degradation caused by system overload, and Little's law is used to identify the system capacity.

VS–RA is evaluated by conducting the experiment, and the results showed that VS–RA provides a significant increase of revenue despite workload variation compared with static and incremental (Inc-Pro) resource allocation. VS–RA can also capture the workload variation and adapt resource allocation much more than the static allocation and the Inc-Pro approach. VS–RA reduces resource usage cost up to 26.8% of the total resource usage and 12.4% more than using Inc-Pro. The limitations of this approach include the following: the process of calculating the 95th percentile from the gamma distribution is a complex process and time consuming, and the queuing-based performance model is application dependent.

5.13. Self-adaptive Neural Fuzzy for Percentile-based Delay Guarantee

The authors [15] addressed the problem of self-provisioning of servers in multi-tier web applications running in a virtualized shared environment. The objective is to provide performance assurance through end-to-end delay guarantees despite a highly dynamic workload and inheriting a multi-tier application complexity.

The server provisioning approach is based on a hybrid of control theory and machine

learning approach. It is capable of fast online self-learning, self-constructing its structure, and adjusting its parameter based on workload variation. As a result, it provides high robustness to time-varying workload, delay target, and server switching delay. The online self-adaptive server provisioning is based on the model-free neural fuzzy controller (NFC) for percentile end-to-end delay guarantee. NFC has a four-layer fuzzy neural network, and each layer provides a specific functionality. The first layer represents the input variables that are considered end-to-end delay errors, and the change in control error. This layer provides the input to the second layer. By contrast, the second layer corresponds to the linguistic value assigned to each input variable; each linguistic value evaluates the input variable through its membership function. Meanwhile, each node in the third layer forms a preconditioned part of fuzzy rules; it multiplies the incoming signal from the precedence nodes and produces the result that represents the strength of the rule. The last layer is responsible for defuzzification of the output to real value to adjust server provisioning.

The experimental results showed that NFC is superior to the rule-based fuzzy controller based on two performance metrics. The first is relative delay deviation, which estimates the square root mean of delay error. Second, temporal violation is the mean of end-to-end violation over frame times. The results demonstrated that the performance during the NFC achieves a small relative delay deviation of 14%, whereas the rule-based fuzzy controller achieves 47%. NFC also shows a violation of 17%, whereas the rule-based fuzzy controller has 38%. NFC improves the performance up to 61% compared with the PI controller for both stationary and non-stationary workloads.

6. Open Challenges and Recommendations

The particular benefits and drawbacks of the theoretical evaluation of computing resource management approaches have been revealed. Based on the insights in the reviewed studies, a number of significant challenges have been detected in regard to current model-based approaches and new trends on resource management development. This section investigates these challenges and trends on resource allocation and briefly presents some recommendations to enhance current research on resource management in multi-tier web applications.

6.1. Open Challenges in Model Based Approaches

Although a large volume of literature on resource management in multi-tier applications exists, significant open challenges are still present. Investigation of reviewed studies finds the following critical challenges on designing model-based approaches for resource management in multi-tier applications.

First, given the performance prerequisites of the hosted applications and virtualized data center specifications, choosing the most suitable model (feedback control, machine learning, linear regression, and queuing network models) to manage resources is important. Despite the numerous recent model-based works in resource management, transparent guidelines on deciding the appreciate model are non-existent. Therefore, further theoretical assessment and investigation of the benefits and limitations of these models are essential.

Second, given a set of performance metrics, such as response time and throughput, suitable inputs to control the target performance metrics should be found. Considering the system complexity and dynamic behavior, identifying a precise relationship between control inputs and target performance that is still valid for different workload regions is difficult. For example, the system throughput can be controlled through CPU entitlement when the system is overloaded, and this relationship becomes invalid when the system is under-loaded [56].

Therefore, determining appropriate control inputs and identifying the relationship between these inputs and target performance under different workloads require more analysis and investigation.

Finally, the system stability under different workload regions should be ensured. Most model-based approaches on feedback control focus on the identification system model and neglect the error brought by workload changing, which is used to identify model parameters. Nevertheless, this error has a significant influence on the stability of the control system, and it may become large enough to cause the instability of the control system. Therefore, ensuring performance regardless of the workload changing is crucial.

6.2. New Research Direction

The reviewed model-based approaches for resource management assume a linear model in the neighborhood of an operating point. However, related work [59] claimed that the relationship between the QoS and the resource entitlement is non-linear. Therefore, a comprehensive research on non-linear control has been conducted [57], [58]. Recently, non-linear control includes feedback linearization, non-linear adaption, and sliding control. The non-linear control model ensures the same QoS as in the linear control model but with fewer resources. However, the designing of a non-linear control model requires rigorous mathematical studies. Despite the difficulties involved in the designing of a non-linear control model, it can help to improve the effectiveness of resource management in multi-tier web applications.

6.3. Research Recommendations

The rule- and model-based approaches for resource management in multi-tier web applications have been investigated. Some open problems related to the application of these approaches remain. These problems include the error caused by the deviation of workload from that used by the identification system model. This section presents some recommendations on how to enhance the effectiveness of the resource management of the reviewed studies.

The investigation and evaluation of this study have revealed that both rule- and model-based approaches are not adequate to provide QoS assurance under workload variations. Some of these approaches are also aggressive toward system dynamics, whereas others are conservative. Knowing that the objective is to provide the QoS assurance with a smaller amount of resource usage costs is essential. One of the methods to tackle this problem and enhance the effectiveness of the applied approaches is to use a hybrid model. In the new model, the model-based approaches provide QoS assurance, whereas the rule-based approaches account for the residual error that results from the inaccuracy of the system model. Therefore, the modern model can adapt to the workload changing faster than the rule- or model-based approaches with less error on the estimation of required resources.

7. Conclusion

Providing performance assurance is crucial for both service providers and application owners. Application owners require their application to produce the desired performance to attract investors, and service providers desire to minimize the resource usage costs to maximize their profit. This study investigates and analyzes resource management approaches in multi-tier web applications. The approaches are classified into two: rule- and model-based; the advantages and disadvantages of these two approaches are illustrated. This study on QoS is

presented and classified into resource provisioning, admission control, and service differentiation. Finally, the crucial challenges, new research direction, and recommendation for providing QoS assurance in multi-tier web applications are introduced.

Table. 2. Literature in Quality of Service

	Adopted approach	QoS management algorithm	Goal	Control level	Evaluation	Benchmark/Dataset
[37]	Lookahead control scheme	Resource provisioning	Maximize profit	Data center & service level	Test-bed	IBM's Trade [53]
[38]	heuristic-based algorithm	Resource provisioning	Minimze resource usage costs	Service level	Simulation & test-bed	TPC-W
[39]	Queueing netwoork theory	Resource provisioning	Minimze resource usage costs	VM level	Test-bed	AOL, UTSlib and Sogou
[36]	proportionalderivative (PD) feedback control	Resource provisioning	Minimize resource usage costs	Service level	Test-bed	synthetic load
[41]	Regression model	Resource provisioning	Provide QoS assurance	Service level	Test-bed	synthetic load
[33]	adaptive feedback controller & queueing network theory	Resource provisioning	Minimize resource usage costs	Service & nodel level	Test-bed	RUBiS
[42]	genetic algorithm	Resource provisioning	QoS guranntee & Cost-effectiveness	Service level	Test-bed	TPC-W [52]
[40]	Queueing network theory	Resource provisioning & Admission control	QoS guranntee	Service level	Test-bed	Rubis [54]
[43]	Queuing network theory & Regression model	Resource provisioning	QoS guranntee	Tier level	Simulation & test-bed	TPC-W
[44]	Regression linear model & optimization theory	Resource provisioning & flowing resources	Reduce resource usage costs	Data center level, Service & node level	prototype	SPECWeb2005 [51]
[45]	Queuing network theory & Adaptive Feedback loop	Admission control	QoS assurance	Service level	Test-bed	TPC-W
[25]	Queuing network theory & Proportional integral controller	Amission control	QoS assurance	Service level	Test-bed	TPC-W
[8]	Fuzzy logic control	Resource provisioning	QoS assurance	Service level	Simulation & test-bed	TPC-W
[46]	Queuing network theory	service differentiation	QoS assurance for prirority services	Tier & node level	Test-bed	IBM's Trade
[47]	Queuing network	Service	QoS assurance for	Service	Test-bed	synthetic load

	theory, optimzation and rule base	differentiation, resource provisioning and Admission control	prirority services & reduce resource usage costs	level		
[48]	MIMO feedback controller	Service differentiation	Maximize profit	Tier level	Test-bed	RUBiS
[15]	Neural Fuzzy controller	Resource provisioning	QoS assurance	Service level	Test-bed	RUBiS
[49]	Queuing network theory & probability	Admission control	QoS assurance	Service level	Test-bed	S-Client [55]
[50]	Queuing newtork	Admission control	QoS assurance	Tier level	Test-bed	TPC-W

References

[1] Douglas and K Barry, “Web Services, Service-oriented Architectures, and Cloud Computing: The Savvy Manager's Guide,” *Morgan Kaufmann Pub*, 2003.

[2] R.T.Fielding and G.Kaiser, “The Apache HTTP Server Project,” *IEEE Internet Computing*, vol.1, no.4, p.88-90, July, 1997. Article (CrossRef Link).

[3] The Apache Jarkarta Project, "Tomcat 6.0.44,” [Online] available at http://jakarta.apache.org/tomcat/.

[4] Java Comm. Process, "The Java Community Process Program,” [Online] http://jcp.org/en/introduction/overview.

[5] Menasc, Daniel A and Almeida, Virgilio AF and Dowdy, Larry W, “Capacity Planning for Web Services: metrics, models, and methods,” *Prentice Hall PTR*, 2002.

[6] Kleinrock and Leonard, “Queuing Systems, Vol. 2: Computer Applications,” *NY: Wiley*, 1976.

[7] Chandra, Abhishek, Weibo Gong, and Prashant Shenoy, "Dynamic resource allocation for shared data centers using online measurements," in *Proc. of Quality of Service—IWQoS 2003*, pp. 381-398. Springer Berlin Heidelberg, 2003. Article (CrossRef Link).

[8] P. Xiong, Z. Wang, G. Jung, and C. Pu, "Study on performance management and application behavior in virtualized environment," in *Proc. of Network Operations and Management Symposium (NOMS)*, 2010 IEEE, pp. 841-844, 2010. Article (CrossRef Link).

[9] B. Urgaonkar, G. Pacifici, P. Shenoy, M. Spreitzer, and A. Tantawi, “An Analytical Model for Multi-Tier Internet Services and Its Applications,” in *Proc. of ACM SIGMETRICS Performance Evaluation Review*, vol. 33, pp. 291–302, 2005. Article (CrossRef Link).

[10] Rackspace, [Online] “http://www.rackspace.com/,” 2012.

[11] Wei, Jianbin, and Cheng-Zhong Xu, "eQoS: Provisioning of client-perceived end-to-end QoS guarantees in web servers," *Computers, IEEE Transactions*, vol. 55, no. 12, pp 1543-1556, 2006. Article (CrossRef Link).

[12] Liu, Xue, Lui Sha, Yixin Diao, Steven Froehlich, Joseph L. Hellerstein, and Sujay Parekh, “Online response time optimization of apache web server,” in *Proc. of Quality of Service—IWQoS 2003*, pp. 461-478, 2003. Article (CrossRef Link).

[13] Zhou, Duanning, and Wayne Wei Huang, "Using a fuzzy classification approach to assess e-commerce web sites: An empirical investigation," *ACM Transactions on Internet Technology (TOIT)*, vol. 9, no. 3, pp. 12, 2009. Article (CrossRef Link).

[14] Diao, Yixin, Joseph L. Hellerstein, and Sujay Parekh, "Optimizing quality of service using fuzzy control," *Management Technologies for E-Commerce and E-Business Applications*, pp. 42-53., 2002. Article (CrossRef Link).

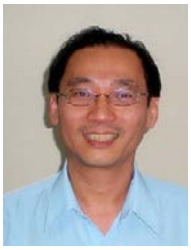
[15] Lama, Palden, and Xiaobo Zhou, "Autonomic provisioning with self-adaptive neural fuzzy control for percentile-based delay guarantee," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 8, no. 2, pp. 9, 2013. Article (CrossRef Link).

[16] Rao, Jia, Yudi Wei, Jiayu Gong, and Cheng-Zhong Xu, "Qos guarantees and service differentiation for dynamic cloud applications," *Network and Service Management, IEEE Transactions*, vol.10, no. 1, pp. 43-55. 2013. Article (CrossRef Link).
[17] Maurer, Michael, Ivona Brandic, and Rizos Sakellariou, "Self-adaptive and resource-efficient SLA enactment for cloud computing infrastructures," in *Proc. of Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*, pp. 368-375, 2012. Article (CrossRef Link).
[18] Lama, Palden, and Xiaobo Zhou, "Efficient server provisioning with control for end-to-end response time guarantee on multitier clusters," *Parallel and Distributed Systems, IEEE Transactions on*, vol.23, no. 1, pp. 78-86, 2012. Article (CrossRef Link).
[19] Wang, Lixi, Jing Xu, Ming Zhao, Yicheng Tu, and Jose AB Fortes, "Fuzzy modeling based resource management for virtualized database systems," in *Proc. of Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS), 2011 IEEE 19th International Symposium on*, pp. 32-42, 2011. Article (CrossRef Link).
[20] Martinez, Jose F., and Engin Ipek, "Dynamic multicore resource management: A machine learning approach," *Micro, IEEE*, vol. 29, no. 5, pp. 8-17, 2009. Article (CrossRef Link).
[21] Tesauro, Gerald, Nicholas K. Jong, Rajarshi Das, and Mohamed N. Bennani, "A hybrid reinforcement learning approach to autonomic resource allocation," in *Proc. of Autonomic Computing, 2006. ICAC'06. IEEE International Conference on*, pp. 65-73, 2006. Article (CrossRef Link).
[22] Xu, Cheng-Zhong, Jia Rao, and Xiangping Bu, "URL: A unified reinforcement learning approach for autonomic cloud management," *Journal of Parallel and Distributed Computing*, vol. 72, no. 2, pp. 95-105, 2012. Article (CrossRef Link).
[23] Bodık, Peter, Rean Griffith, Charles Sutton, Armando Fox, Michael Jordan, and David Patterson, "Statistical machine learning makes automatic control practical for internet datacenters," in *Proc. of the 2009 conference on Hot topics in cloud computing*, pp. 12-12, 2009.
[24] Gong, Zhenhuan, Xiaohui Gu, and John Wilkes, "Press: Predictive elastic resource scaling for cloud systems," in *Proc. of Network and Service Management (CNSM), 2010 International Conference on IEEE*, pp. 9-16, 2010. Articale (CrossRef Link).
[25] Kamra, Abhinav, Vishal Misra, and Erich M. Nahum, "Yaksha: A self-tuning controller for managing the performance of 3-tiered web sites," in *Proc. of Quality of Service, 2004. IWQOS 2004. Twelfth IEEE International Workshop on*, pp. 47-56, 2004. Article (CrossRef Link).
[26] Liu, Xue, Jin Heo, Lui Sha, and Xiaoyun Zhu, "Adaptive control of multi-tiered web applications using queueing predictor," in *Proc. of Network Operations and Management Symposium, 2006. NOMS 2006. 10th IEEE/IFIP*, pp. 106-114, 2006. Article (CrossRef Link).
[27] Hellerstein, Joseph L., Yixin Diao, Sujay Parekh, and Dawn M. Tilbury, "Feedback control of computing systems," *John Wiley & Sons*, 2004. Article (CrossRef Link).
[28] Lu, Chenyang, Ying Lu, Tarek F. Abdelzaher, John A. Stankovic, and Sang Hyuk Son, "Feedback control architecture and design methodology for service delay guarantees in web servers," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 17, no. 9, pp. 1014-1027, 2006. Article (CrossRef Link).
[29] Lu, Ying, Tarek Abdelzaher, Chenyang Lu, Lui Sha, and Xue Liu, "Feedback control with queueing-theoretic prediction for relative delay guarantees in web servers," in *Proc. of Real-Time and Embedded Technology and Applications Symposium, 2003. Proceedings. The 9th IEEE*, pp. pp. 208-217, 2003. Article (CrossRef Link).
[30] Abdelzaher, Tarek, Ying Lu, Ronghua Zhang, and Dan Henriksson, "Practical application of control theory to web services," in *Proc. of American Control Conference, 2004. The 2004*, IEEE, vol. 3, pp. 1992-1997, 2004. Article (CrossRef Link).
[31] Chandra, Abhishek, Weibo Gong, and Prashant Shenoy, "Dynamic resource allocation for shared data centers using online measurements," in *Proc. of Quality of Service—IWQoS 2003, Springer Berlin Heidelberg*, pp. 381-398, 2003. Article (CrossRef Link).
[32] Urgaonkar, Bhuvan, Giovanni Pacifici, Prashant Shenoy, Mike Spreitzer, and Asser Tantawi, "An analytical model for multi-tier internet services and its applications," in *Proc. of ACM SIGMETRICS Performance Evaluation Review*, vol. 33, no. 1, pp. 291-302. ACM, 2005.

Article (CrossRef Link).
[33] Xiong, Pengcheng, Zhikui Wang, Simon Malkowski, Qingyang Wang, Deepal Jayasinghe, and Calton Pu, "Economical and robust provisioning of n-tier cloud workloads: A multi-level control approach," in *Proc. of Distributed Computing Systems (ICDCS), 2011 31st International Conference on*, pp. 571-580. IEEE, 2011. Article (CrossRef Link).
[34] Padala, Pradeep, Kai-Yuan Hou, Kang G. Shin, Xiaoyun Zhu, Mustafa Uysal, Zhikui Wang, Sharad Singhal, and Arif Merchant, "Automated control of multiple virtualized resources," in *Proc. of the 4th ACM European conference on Computer systems*, pp. 13-26. ACM, 2009. Article (CrossRef Link).
[35] Tang, Chunqiang, Malgorzata Steinder, Michael Spreitzer, and Giovanni Pacifici, "A scalable application placement controller for enterprise data centers," in *Proc. of the 16th international conference on World Wide Web*, pp. 331-340. ACM, 2007. Article (CrossRef Link).
[36] Urgaonkar, Bhuvan, Prashant Shenoy, Abhishek Chandra, Pawan Goyal, and Timothy Wood, "Agile dynamic provisioning of multi-tier internet applications," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 3, no. 1, pp. 1, 2008. Article (CrossRef Link).
[37] Kusic, Dara, Jeffrey O. Kephart, James E. Hanson, Nagarajan Kandasamy, and Guofei Jiang, "Power and performance management of virtualized computing environments via lookahead control," *Cluster computing*, vol. 12, no. 1, pp. 1-15, 2009. Article (CrossRef Link).
[38] Han, Rui, Moustafa M. Ghanem, Li Guo, Yike Guo, and Michelle Osmond, "Enabling cost-aware and adaptive elasticity of multi-tier cloud applications," *Future Generation Computer Systems*, vol. 32, pp. 82-98, 2014. Article (CrossRef Link).
[39] Jiang, Jing, Jie Lu, Guangquan Zhang, and Guodong Long, "Optimal cloud resource auto-scaling for web applications," in *Proc. of Cluster, Cloud and Grid Computing (CCGrid), 2013 13th IEEE/ACM International Symposium on*, pp. 58-65, 2013. Article (CrossRef Link).
[40] Ashraf, Adnan, Benjamin Byholm, Joonas Lehtinen, and Ivan Porres, "Feedback control algorithms to deploy and scale multiple web applications per virtual machine," in *Proc. of Software Engineering and Advanced Applications (SEAA), 2012 38th EUROMICRO Conference on*, pp. 431-438, 2012. Article (CrossRef Link).
[41] Iqbal, Waheed, Matthew N. Dailey, David Carrera, and Paul Janecek, "Adaptive resource provisioning for read intensive multi-tier applications in the cloud," *Future Generation Computer Systems*, vol. 27, no. 6, pp. 871-879, 2011. Article (CrossRef Link).
[42] Mi, Haibo, Huaimin Wang, Gang Yin, Yangfan Zhou, Dianxi Shi, and Lin Yuan, "Online self-reconfiguration with performance guarantee for energy-efficient large-scale cloud computing data centers," in *Proc. of Services Computing (SCC), 2010 IEEE International Conference on*, pp. 514-521, 2010. Article (CrossRef Link).
[43] Zhang, Qi, Ludmila Cherkasova, Ningfang Mi, and Evgenia Smirni, "A regression-based analytic model for capacity planning of multi-tier applications," *Cluster Computing*, vol. 11, no. 3, pp. 197-211, 2008. Article (CrossRef Link).
[44] Song, Ying, Hui Wang, Yaqiong Li, Binquan Feng, and Yuzhong Sun, "Multi-tiered on-demand resource scheduling for VM-based data center," in *Proc. of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*, pp. 148-155, 2009. Article (CrossRef Link).
[45] Liu, Xue, Jin Heo, Lui Sha, and Xiaoyun Zhu, "Queueing-model-based adaptive control of multi-tiered web applications," *Network and Service Management, IEEE Transactions on*, vol. 5, no. 3, pp. 157-167, 2008. Article (CrossRef Link).
[46] Diao, Yixin, Joseph L. Hellerstein, Sujay Parekh, Hidayatullah Shaikh, and Maheswaran Surendra, "Controlling quality of service in multi-tier web applications," in *Proc. of Distributed Computing Systems, 2006. ICDCS 2006. 26th IEEE International Conference on*, pp. 25-25, 2006. Article (CrossRef Link).
[47] Wang, Xiaoying, Zhihui Du, Yinong Chen, and Sanli Li, "Virtualization-based autonomic resource management for multi-tier Web applications in shared data center," *Journal of Systems and Software*, vol. 81, no. 9, pp. 1591-1608, 2008. Article (CrossRef Link).

[48] Liu, Xue, Xiaoyun Zhu, Pradeep Padala, Zhikui Wang, and Sharad Singhal, "Optimal multivariate control for differentiated services on a shared hosting platform," in *Proc. of Decision and Control, 2007 46th IEEE Conference on*, pp. 3792-3799, 2007. Article (CrossRef Link).
[49] Cao, Jianhua, Mikael Andersson, Christian Nyberg, and Maria Kihl, "Web server performance modeling using an M/G/1/K* PS queue," in *Proc. of Telecommunications, 2003. ICT 2003. 10th International Conference on*, vol. 2, pp. 1501-1506, 2003. Article (CrossRef Link).
[50] Liu, Xue, Jin Heo, and Lui Sha, "Modeling 3-tiered web applications," in *Proc. of Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2005. 13th IEEE International Symposium on*, pp. 307-310, 2005. Article (CrossRef Link).
[51] Standard Performance Evaluation Corporation, [Online] https://www.spec.org/web2005/.
[52] Transaction Processing performance Council, [Online], http://www.tpc.org/tpcw/
[53] BM WebSphere Software, http://www.ibm.com/software/webservers/appserv/benchmark3.html.
[54] Rubis, [Online] http://rubis.ow2.org/.
[55] Banga, Gaurav, and Peter Druschel, "Measuring the Capacity of a Web Server," in *Proc. of USENIX Symposium on Internet Technologies and Systems*, pp. 61-71. 1997.
[56] Wang, Zhikui, Xiaoyun Zhu, and Sharad Singhal, "Utilization and SLO-based control for dynamic sizing of resource partitions," *Ambient Networks*, pp. 133-144, 2005. Article (CrossRef Link).
[57] Slotine, Jean-Jacques E., and Weiping Li, *Applied nonlinear control*. Vol. 60. Englewood Cliffs, NJ: Prentice-Hall, 1991.
[58] Arjan, S. *L2-gain and passivity techniques in nonlinear control*, New York, 2000.
[59] Wang, Zhikui, Yuan Chen, Daniel Gmach, Sharad Singhal, Brian J. Watson, Wilson Rivera, Xiaoyun Zhu, and Chris D. Hyser, "Appraise: application-level performance management in virtualized server environments," *Network and Service Management, IEEE Transactions on*, vol. 6, no. 4, pp. 240-254, 2009. Article (CrossRef Link).

Mohamed Ghetas is a PhD candidate from the School of Computer Sciences, (USM) Universiti Sains Malaysia. He received his M.Sc. degrees from Universiti Sains Malaysia in 2013. His research interests include to cloud computing and intelligence systems.



Huah-Yong Chan is an associate professor in the School of Computer Sciences, Universiti Sains Malaysia (USM), Malaysia. He is a head of cloud computing lab at USM. He received his Ph.D, degree from the Université de Franche-Comté, France, in 1999. He is actively involved in grid computing research activities, both, at the national and international level. His research spans from cloud computing, to a more specific issues such as resource allocation, load balancing, software agents, middleware engineering and cloud computing.

Putra Sumari is an associate professor and lecturer at School of Computer Science, Universiti Sains Malaysia, He received his MSc and PhD in 1997 and 2000 from Liverpool University, England, member of ACM and IEEE, program committee and reviewer of several journals and international Conferences. He has published more than hundred papers including journal and conferences. His research areas are multimedia communication, content distributing network, and cloud computing.