ISSN:0975-9646

A Novel Fuzzy Controller for Dynamic Cloud Applications

R. Nandhiniprabha, G. Sudhakar

***Abstract*— Cloud elasticity allows dynamic resource provisioning in concert with actual application demands. Self Tuning fuzzy control has been applied with success to resource allocation. However, cloud dynamics make the design of the accurate and stable resource controller challenging, especially when client request increased gradually. To address these challenges, we introduced novel fuzzy controller (NFC) approach with neural concept and used scheduling algorithm. It supports adaptive multi-objective and service differentiation. Experimental results on representative server workloads show that NFC approach outperforms popular controllers such as Kalman filter, ARMA and, Adaptive PI in the control of CPU, memory, and disk bandwidth resources under both static and dynamic workloads.**

***Index Terms*— Quality of service, cloud management, service differentiation, fuzzy control.**

I. Introduction

Cloud Computing is a paradigm for enabling remote, on demand access to a set of configurable computing resources. This model aims to provide hardware and software services to customers, while minimizing human efforts in terms of service installation, configuration and maintenance, for both cloud provider and cloud customer. As server virtualization grows increasingly popular and a mature, hosting enterprise application in a cloud has become an attractive solution for scalability and cost efficiency. Applications running within virtual machines have on demand access to compute resources in response to increased application loads. Thus virtual machines should be dynamically provisioned to match actual application demands, rather than the peak requirement. However, these demands are difficult to estimate due to time-varying and diverse workload. Then client perceived quality of service should still be maintained in the presence of background dynamic resource provisioning. These observations call for an effective approach that automates the resource allocation for cloud users.

There are many control approaches that have been applied with success to resource allocation in physical servers; see[2], [3], [4], [5], [6] for examples. Recent studies have focused on the application of control approaches for the allocation of virtualized resources in cloud [7], [8], [9], [10], [11]. The cloud adds new challenges to the QoS-oriented resource allocation, in addition to workload dynamics. Many existing work used indirect metrics such as workload arrival rate and CPU utilization instead of considering application level performance as measured output.

To address the issue of the lack of an accurate server model, the work in [9], [11] applied adaptive control approaches based on model approximation. However, these approaches pose limitations on how fast the workload and the system behavior can change [12]. In [13], we developed a two-layer novel fuzzy control (NFC) approach for QoS assurance in web servers with respect to response time. In this paper, we extend the NFC approach to multiple resource allocations in virtualized environments by introducing a novel fuzzy control (NFC) approach. In comparison with other popular controllers, it shows better adaptability and stability. However, these metrics usually behave nonlinearly with respect to resource allocations and are highly dependent on the characteristics of workload, as well as server capacity. This nonlinearity poses challenges to design a stable and accurate controller.

To evaluate the performance of NFC and the DynaQoS framework, we have built a cloud testbed based on a Xen environment. We conducted experiments to dynamically control the allocation of CPU, memory, and disk bandwidth resources to three representative cloud applications. These applications include a cluster-based E-Commerce website (TPC-W [14]), an in-memory key-value store (Memcached [11]), and a video streaming server (Darwin streaming server [8]). For comparison with NFC, we also implemented three popular controllers within the DynaQoS framework: a model-independent Adaptive PI controller and two controllers based on local model approximation: Kalman filter and ARMA controllers. In summary, this paper makes the following contributions:

1) Proposed a novel framework, DynaQoS, to provide QoS-guaranteed automatic resource management to cloud applications. The framework also includes the support for multiple-objective control and service differentiation.
2) Implementing the DynaQoS framework and demonstrating its effectiveness on a Xen-based cloud testbed.

The rest of this paper is organized as follows. Section II discusses the design objectives and the challenges in automatic cloud resource management. Section III and Section IV elaborate the key designs and implementation of framework, respectively. Related work is presented in Section V. We conclude this paper in Section VI.

II. Overview, Assumption And Goals

In this section, we review the design objectives of a resource controller and discuss the challenges in a cloud environment.

A. *Design Objectives*

For cloud users, leasing virtual servers from cloud is advantageous over buying physical machines only if they leverage the elasticity of cloud and dynamically maintained the virtual resources to a level that matches actual application demands [15]. Feedback control is a promising method for automatic resource allocation. A resource controller should achieve diverse control objectives, maintain high resource utilization, while still guarantee application-level QoS and realize service differentiation. In the following, we outline the design objectives of a resource controller:

- **Precise control of multiple user-defined metrics**. The controller should transparently translate user defined high level control metrics to low-level resource requirements and allocate resources in a way that minimizes the error between the measured output and the desired output. These metrics include but are not limited to performance metrics (e.g., response time and throughput), expenditure metrics (e.g., dollars per hour), and energy consumption metrics (e.g., Joule per hour). Moreover, the controller should also support the simultaneous control of multiple metrics if necessary. For example, a cloud user may request a response time target of 1 second together with a power budget of 250 watt.
- **QoS guarantee and differentiation**. The controller should ensure that application level QoS is guaranteed in the presence of dynamic resource control. It requires that the controller be responsive to QoS violations and stable during oscillations. When resources are constrained, the controller should provide differentiated services to different service classes.
- **Reduced rejected request.**. Besides meeting the stated control and QoS objectives, the resource controller should also maintain the utilization of resources at high levels. This avoids the waste in idle resources and leads to savings in the leasing cost.

B. *The Challenges*

To build a resource controller realizing a high-level ob-jective, a mathematical model that captures the relationship between the allocated resource and the high-level metric is necessary. Given the model, any deviation of the high-level metric from the desired value can be corrected by applying adjustments in the resource allocation. However, the determination of the system model in a dynamic cloud environment is not trivial. Workload and cloud dynamics make the identification of system models difficult. In the following, we discuss the causes of dynamic capacity and show that the uncertainties affect application modeling.

1) Performance interference: In a cloud environment, mul-tiple cloud users share the same infrastructure. Although server virtualization helps realizing performance isolation to some extent, VMs from different cloud users may still have chances to interfere with each other. We show that the involvement of the centralized virtualization layer in CPU, memory and I/O device virtualization causes uncertainties in VM resource allocation [14]. Rogue applications may deprive the resources in the hypervisor and incur significant performance degradation to other applications. For some applications, their performance is dependent on the characteristics of co-hosting VMs. For example, the actual CPU performance relies on the memory intensity of co-running workloads that share the last-level cache and the actual disk performance depends on the sequentiality of other jobs. Thus, application models obtained offline are likely to be inaccurate online with interference from other applications.

2) The non-uniformity of cloud resources: In this subsection, we show that even without resource contention from others, cloud users may still see variations in the capacity of their applications due to the inherent non-uniformity of cloud hardware.

3) High resource utilization: Besides uncertainties from the underlying cloud hardware, dynamics in VMs' capacity can also come from market-based accesses to additional resources.

III. NOVEL DYNAQOS FRAMEWORK

DynaQoS is composed of two layers of controllers. The first layer is a group of Novel fuzzy controllers (NFC) that control individual objectives. During each control interval, a NFC queries the corresponding QoS profile manager for the reference value of the controlled metric. A QoS monitor periodically reports the achieved value of the metric. The metrics to be controlled can be conventional application-level performance metrics such as response time or throughput; or any user-defined high-level metric The NFC takes the difference between the reference value and the achieved one as well as the change of the error as its inputs and outputs a resource request to the second layer weighted scheduler.

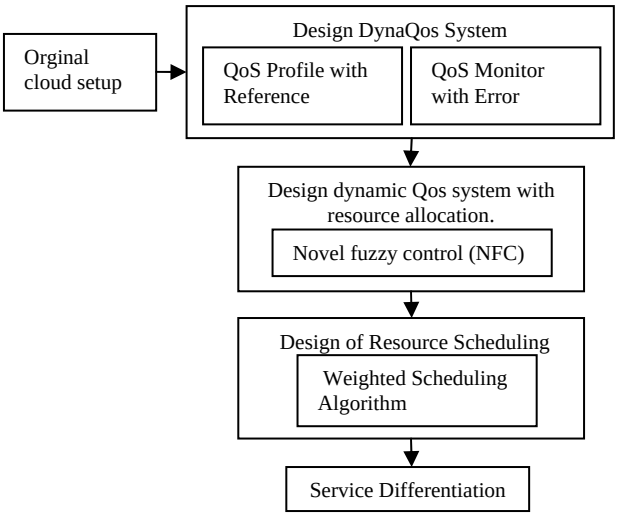

Fig 1. The structure of the NFC

When there are multiple control objectives, the second layer weighted scheduler aggregates the resource requests

from individual NFCs and forms a unified resource request. The aggregation of individual requests is based on the weights (gain) of each NFC when determining the final request. The gains are dynamically adjusted according to the control errors of NFCs. Service differentiation is necessary if multiple service classes exist and the aggregated resource demand is beyond the available capacity.

Design of the Novel Fuzzy Controller: The fuzzy controller only defines the basic control rules according to the inputs of $e(k)$ and $e(k)$. It outputs the sign and magnitude of the allocation adjustment $u(k)$. With cloud dynamics, there could be a lot of fluctuations in the control effect. The fuzzification component translates the inputs into corresponding certainty in numeric values of the membership function. The interference mechanism is to determine which rules should be activated and what are the conclusions. Based on the outputs of the interference mechanism, the defuzzification component calculates the fuzzy controller output.

To measure the performance of a controller, we define a metric, relative deviation R(e), based on root mean error.

$$R(e) = \frac{\sqrt{\sum_{k=1}^{n} e(k)^2/n}}{r(k)}$$

The above mentioned works mainly address the problem from QoS routing perspective, which cannot be applied directly in virtualization-based Cloud computing for QoS guaranteed service composition. This is because network virtualization brings new challenges that a virtualized service network may actually span a great scale of heterogeneous networks with diverse network resources in different domains. In this paper, we investigate an optimization issue for Cloud service provisioning, that is, QoS-aware service provisioning upon network virtualization in Cloud computing.

To address the problem of process delay and control inaccuracies, fuzzy control rules also need to be tuned based on current conditions. The weighted scheduler aggregates the resource request from individual NFCs and forms a unified resource request. The aggregation of individual request is based on the weights of each NFC when determining the final request. The weights are dynamically adjusted according to the control errors and service differentiation.

To provide QoS guarantees, we consider service differentiation to be initiated by individual service classes. When resource contentions are detected, the service class with a lower priority would adapt its SLO (e.g., a response time target) to a lower level. By setting different control objectives, the premium class will receive more resources than the basic class while the basic class will not be starved for resources by maintaining a degraded level of service. We enforce strict priorities between classes. That is the class with a higher priority adapts to a lower level only when the lower priority classes have reached their minimum service levels. To detect resource contentions, DynaQoS follows a simple heuristic of tracking the control performance.

IV. IMPLEMENTATION MODEL

C. Implementation of Novel DynaQoS

QoS monitor. We measured application-level performance at the client side of each cloud application. Generally, we modified the workload generators to maintain logs of finished requests. We wrote utility programs to parse the logs and calculate the average performance for every control interval. We used response time, throughput, and play rate as the control outputs for TPC-W, Memcached, and Darwin streaming server, respectively.

QoS profile manager. Each service class works with a QoS profile manager to determine the control objective. The control objectives are specified in terms of a set of desired control outputs with different levels. For service differentiation, the profile manager also sets the number of SLO violations that can be tolerated by a class before a target adaptation is needed. For service differentiation in Section V-C, we only considered the service differentiation in TPC-W and two classes: *Premium* and *Basic*. They both have three levels of SLO specified in terms of response times, *{*1s, 5s, 10s*}*, and with adaptation thresholds: 10 and 30 violations, respectively.

Novel fuzzy controller. NFC has been implemented as a set of user-level daemons in the virtual host (i.e., dom0 in a Xen environment). It takes the measured application-level performance (from QoS monitor) and the performance objective (QoS profile manager) as input and outputs the re-source adjustment to Xen's management interface. If multiple control objectives exist, two or more NFCs form a unified request. The control interval is set to 30 seconds for all the experiments.

To compare the performance of different controllers, we take the performance of NFC as a baseline and define the performance difference between NFC and other controllers as

$$Perf\ Diff = \frac{R(e)_{other} - R(e)_{NFC}}{R(e)_{NFC}}$$

Resource Utilization. CPU resources are allocated to each DB VM via Xen Credit Scheduler in terms of cap values. A *cap* value represents the upper limit of CPU time can be consumed by a VM. For a virtual cluster with 4 VMs and each with 4 cores, the CPU allocation can be in the range of [1, 1600]. The CPU time is allocated to individual virtual clusters. We assume good load balancing by the Tomcat balancer, thus distribute CPU cap values uniformly within the cluster. All VMs are given the same weight during allocation.

Control theory has recently been applied to computer systems for resource management and performance control [10], [12], [13], [14]. The application areas include web server performance guarantees [1], dynamic adjustment of the cache size for multiple request classes [15], guaranteed relative delays in web servers by connection scheduling [9], CPU and memory utilization control in web servers [5] and to adjust the resource demands of virtual machines based on resource availability.

V. RELATED WORK

Control theory has recently been applied to computer systems for resource management and performance control [10], [11], [13]. The application areas include web server performance guarantees [1], dynamic adjustment of the cache size for multiple request classes [15], guaranteed relative delays in web servers by connection scheduling [14], CPU and memory utilization control in web servers [9] and to adjust the resource demands of virtual machines based on resource availability.

Provisioning of QoS guarantees has been an active research topic. Early work focused on provisioning service guaran-tees or differentiation under fixed capacity. Methods such as queuing-theoretic analysis, traditional feedback control, and adaptive control have been studied extensively. In [28], the authors assumed a *G/G/*1 queuing model to guide the resource allocation. However, this approach depends on the parameter estimation of the model, which is difficult to ob-tain without understanding the system internals. Due to the absence of the knowledge of underlying systems, traditional linear feedback control was applied to control the resource allocation in web servers [2], [3], [4]. Because the behavior of a web server changes continuously, the performance of the linear feedback control is limited. More recent work applied adaptive control [5], [6] and machine learning [15] to address the issue of the lack of an accurate server model. Although these approaches provide better performance than non-adaptive feedback control approaches, they did not address the problem of process delay in resource allocation.

Our previous work [15] used an adaptive fuzzy control approach without the assumption of a server model to address the process delay in resource allocation. In this work, DynaQoS improves our eQoS work [15] in several ways. In our prior work, we have developed a suite of dynamic allocation techniques for virtualized servers, including adaptive control of resource allocation under overload conditions[9], nonlinear adaptive control for dealing with nonlinearity and bimodal behavior of the system [10], and nested control for a better tradeoff between resource utilization and application-level performance [12]. These approaches are suitable for applications that are hosted inside a single virtual machine. In this paper, we present a dynamic resource allocation system for multi-tier applications with individual components distributed in different virtual machines. Our fuzzy control-based approach does rely on the understanding of underlying systems and deals with nonlinearities. Our work is different because NFC directly operates on the control error and the change of the error. Thus, it avoids the computation of fuzzy rules for adaptability.

VI. CONCLUSION

This paper has presented the novel fuzzy control approach for the resource allocation. Based on the fuzzy controller, we developed Novel DynaQoS, which supports adaptive multi-objective resource allocation. This framework and weighted scheduler demonstrated its effectiveness in the simultaneous control of performance, power and service differentiation. It works well when the client demand increased gradually, and it improves the resource management and reduced the rejected request.

REFERENCES

[1] J. L. Hellerstein, Y. Diao, S. Parekh, and D. M. Tilbury, *Feedback Control of Computing Systems*. John Wiley & Sons, 2004.

[2] T. F. Abdelzaher, K. G. Shin, and N. Bhatti, "Performance guarantees for web server end-systems: a control-theoretical approach," *IEEE Trans. Parallel Distrib. Syst.*, 13 January, 2002.

[3] C. Lu, T. F. Abdelzaher, J. A. Stankovic, and S. H. Son, "A feedback control approach for guaranteeing relative delays in web servers," in *2001 RTAS*.

[4] P. P. Renu, P. Pradhan, R. Tewari, S. Sahu, A. Ch, and P. Shenoy, "An observation-based approach towards self-managing web servers," in *2002 IWQoS*.

[5] A. Kamra, V. Misra, and E. M. Nahum, "Yaksha: a self-tuning controller for managing the performance of 3-tiered web sites," in *2004 IWQoS*.

[6] M. Karlsson, C. T. Karamanolis, and X. Zhu, "Triage: performance isolation and differentiation for storage systems," in *2004 IWQoS*.

[7] D. Kusic, J. O. Kephart, J. E. Hanson, N. Kandasamy, and G. Jiang, "Power and performance management of virtualized computing envi-ronments via lookahead control," in *2008 ICAC*.

[8] R. Wang, D. M. Kusic, and N. Kandasamy, "A distributed control framework for performance management of virtualized computing en-vironments," in *2010 ICAC*.

[9] E. Kalyvianaki, T. Charalambous, and S. Hand, "Self-adaptive and self-configured CPU resource provisioning for virtualized servers using Kalman filters," in *2009 ICAC*.

[10] P. Padala, K. G. Shin, X. Zhu, M. Uysal, Z. Wang, S. Singhal, A. Merchant, and K. Salem, "Adaptive control of virtualized resources in utility computing environments," in *2007 EuroSys*.

[11] P. Padala, K.-Y. Hou, K. G. Shin, X. Zhu, M. Uysal, Z. Wang, S. Singhal, and A. Merchant, "Automated control of multiple virtualized resources," in *2009 EuroSys*.

[12] F. Zhou, M. Goel, P. Desnoyers, and R. Sundaram, "Scheduler vulner-abilities and attacks in cloud computing," arXiv:1103.0759v1 [cs.DC], Mar. 2011.

[13] J. Wei and C.-Z. Xu, "EQoS: provisioning of client-perceived end-to-end QoS guarantees in web servers," *IEEE Trans. Computer*, vol. 55, 2006.

[14] J. Rao, X. Bu, C.-Z. Xu, L. Wang, and G. Yin, "VCONF: a reinforce-ment learning approach to virtual machines auto-configuration," in *2009 ICAC*.

[15] J. Rao and C.-Z. Xu, "Online measurement the capacity of multi-tier websites using hardware performance counters," in *2008 ICDCS*.