Honoring SLAs on cloud computing services: a control perspective

Christos A. Yfoulis and Anastasios Gounaris

***Abstract*— This work contains a short survey of recent results in the literature with a view to opening up new research directions for the problem of honoring SLAs on cloud computing services. This is a new problem that has attracted significant interest recently, due to the urgent need for providers to provide reliable, customized and QoS guaranteed computing dynamic environments for end-users as agreed in contracts on the basis of certain Service Level Agreements (SLAs). Honoring SLAs is a multi-faceted problem that may involve optimal use of the available resources, optimization of the system's performance and availability or maximization of the provider's revenue and it poses a significant challenge for researchers and system administrators due to the volatile, huge and unpredictable Web environments where these computing systems reside. The use of algorithms possessing run-time adaptation features, such as dynamic resource allocation, admission control and optimization becomes an absolute must. As a continuation of the recent successful application of control theory concepts and methods to the computing systems area, our survey indicates that the problem of honoring SLAs on cloud computing services is a new interesting application for control theory and that researchers can benefit significantly from a number of well-known modern control methodologies, such as hybrid, supervisory, hierarchical and model predictive control.**

I. Introduction

Cloud computing is emerging as a new computing paradigm and is being driven by many well known IT providers (Amazon, Google and Yahoo!) and vendors (HP, IBM, Intel and Microsoft). Although we still lack a widely accepted definition for cloud computing, it is an Internet-based ("cloud") development and use of computer technology ("computing") that promises to offer flexible dynamic IT infrastructures, QoS guaranteed computing environments and configurable software services [1].

Cloud computing builds on top of several other technologies, i.e. distributed computing, grid computing, utility computing and autonomic computing, and it can be envisaged as a natural step forward from the grid-utility model. In the heart of cloud computing infrastructure we find a group of reliable services delivered through powerful *data computing centers* that are based on modern *virtualization* technologies and related concepts such as component-based system engineering, orchestration of different services through *workflows* and *service-oriented architectures* (SOAs) [2]. The *Cloud* constitutes a single point of access for all services which are available anywhere in the world, on the basis of commercial contracts that guarantee satisfaction of the QoS requirements of customers according to specific *service level agreements (SLAs)*.

The purpose of using SLAs is to define a formal basis for performance and availability the provider guarantees to deliver. SLA contracts record the level of service, specified by several attributes such as availability, serviceability, performance, operation, billing or even penalties in the case of violation of the SLA. Also, a number of *performance-related metrics* are frequently used by Internet Service Providers (ISPs), such as service response time, data transfer rate, round-trip time, packet loss ratio and delay variance.

Often, providers and customers negotiate *utility-based* SLAs that determine the cost and penalties based on the achieved performance level. A resource allocation management scheme is usually employed with a view to maximizing overall profit (utility) which includes the revenues and penalties incurred when QoS guarantees are satisfied or violated, respectively. Usually, step-wise utility functions are used where the revenue depends on the QoS levels in a discrete fashion, as shown in Figure 1. Finally, a concept central to the development of cloud computing which constitutes an integrated view of service-based activities is provided by the idea of a *workflow*, which represents a series of structured activities and computations that arise in IT services, and has been often represented by a directed graph connecting both loosely and tightly coupled processing components.

The contributions of this work are summarized as follows: (i) it introduces a representative problem in accordance with the modern commercial cloud computing paradigm; (ii) it contains a short review of relevant works in the literature, with a special attention to techniques based on control theory; (iii) it investigates the possibility of designing efficient control systems based on advanced control methodologies for solving the problem in question.

The remainder of this article is structured as follows. Section II describes the main problem formally. The presentation of a number of control theory-based approaches in the literature that are related to our problem takes place in Section III, whereas in Section IV some promising advanced control methodologies are proposed. Section V concludes the article.

II. Problem description

In this section we begin with the formulation of a representative problem on cloud computing as the reference point for our discussion.

Christos A. Yfoulis is with the Automation Department, ATEI of Thessaloniki, Thessaloniki, Greece. E-mail: cyfoulis@teithe.gr

Anastasios Gounaris is with the Computer Science Department, Aristotle University of Thessaloniki, Thessaloniki, Greece. E-mail: gounaria@csd.auth.gr

Problem 1 *Let us assume that a service owner provides (1) a cluster of M machines and (2) a set of services, which can be combined together to form a workflow. Each service has multiple instantiations, each with different QoS characteristics. Multiple users share this infrastructure, and they can submit workloads consisting of multiple workflow requests under the condition that the owner has accepted the SLA agreement that the users have proposed. The SLAs are in the form of Figure 1. The work focuses on two aspects:*

- *performing admission control, i.e., decide which SLAs should be accepted;*
- *maximizing the owner's profit. Profits are generated when SLA agreements are honored, i.e., when workload execution completes on time; otherwise, penalties are incurred.*

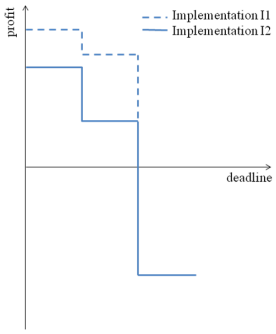

Fig. 1. Example SLA referring to a workflow request that can be implemented in two ways, each having a different level of QoS and yielding different profits if completed on time.

To achieve the latter, (1) the number of machines allocated to each workload and (2) the implementation of each service are continuously adapted at runtime.

Further assumptions are as follows:

- *at least one machine is allocated to each workload whose SLA has been accepted;*
- *the execution time of each service implementation follows a known distribution;*
- *the workload arrival rate follows a known distribution;*
- *service execution is non- pre-emptive; however workload execution is pre-emptive (i.e, the number of machines allocated to a single workload can vary at runtime).*

■

Note that there are many ways by which this formulation may be modified or extended, e.g. to the case of *virtualized* resources or *multi-tier* architectures, or when different SLA criteria and/or tuning parameters apply.

There are many papers in the literature where constrained optimization problems are solved with a view to maximizing the revenue of the host provider using a network flow model, based upon queueing theory formulas, e.g. [3], [4], [5], [6]. These works do not utilize any control theory concepts and provide techniques for the solution of similar but less complicated problems compared to Problem 1. Research efforts to find a solution to Problem 1 could use similar models, assumptions and ideas as a starting point. In the next section we focus on control theoretical techniques proposed in the literature that are related to Problem 1.

III. Approaches employing control theoretical solutions

A. Introduction

There are many different types of control solutions in the literature for related problems that could be useful in our context. The distinct characteristics of each one of them could be attributed to a couple of issues, stated below.

1) **Control objectives***:* The *control objective* may be a) *regulation* of QoS metrics of interest – e.g. response time (delay) or CPU utilization– to desired known reference values predefined by the user or the administrator, b) *optimization* of QoS metrics to unknown optimal or sub-optimal values or c) *disturbance rejection* due to the presence of unpredictable workload changes, which is the case in any computing system. Of course, in many cases a combination of the previous objectives may be needed, and in fact we believe that a well-founded solution to a computing problem that is also optimal, reliable and of practical merit has to incorporate all three objectives, as well as deal with the presence of *hard* constraints such as physical resources bounds or *soft* constraints such as response time deadlines, specified e.g. in SLAs.

2) **Metrics and Adaptation mechanisms***:* There are several output metrics employed in QoS control design for a variety of computer systems and software. These include *system-level* metrics, such as CPU and memory utilization, cache hit ratio, server queue length etc., *application-level* metrics such as response time and throughput, or *business-level* metrics such as profits in SLAs.

However, in a general setting, enforcing SLAs may be equivalent to minimizing a certain objective function, which may involve several different QoS metrics, such as response time, utilization, maintenance cost, revenue etc. or a user-defined weighted sum of some of them, which corresponds to a trade-off among conflicting goals. There are three main *adaptation mechanisms* : a) *admission control*, i.e. to reject some customer requests in order to avoid overload or recover from overload conditions that seriously affect the performance of customers already connected, b) *resource reallocations*, i.e. a mechanism for dynamic resource allocations reflecting the dynamic conditions experienced in the presence of time-varying and unpredictable workloads, and c) *real-time scheduling* of tasks.

We could also consider additionally *load balancing* and *content adaptation*. It is obvious that there may be frameworks that could include combinations of some or all of these mechanisms, and an important issue then is how these strategies can be synchronized and work in a coordinated manner without conflicts between them. In other cases some of the mechanisms –e.g. load balancing or real-time scheduling– may already exist in a certain implementation (and even cause time-varying traffic) and the goal may be to design

another controller on top of them to optimize the overall performance.

B. Main research directions

On the basis of the previously mentioned issues, we continue with a more detailed presentation of the relevant works in the literature, and comment on how these ideas could be useful ingredients in relation to our problem.

1) Hill climbing heuristic optimization: In [7], profit-oriented feedback control solutions are proposed in e-commerce services specified by SLAs. A controller is employed, that automates the admission control decisions in a way that balances the loss of revenue due to rejected work against the penalties incurred if admitted work has excessive response times.

Fuzzy controllers implement a hill climbing logic to maximize the total profit, which under certain assumptions can be shown to be a concave backward function of the tuning admission control parameter. Fuzzy control has been shown to be robust to changes in workloads and values of SLA parameters and can handle the stochastics reasonably well. Other similar hill climbing techniques (*extremum control*) with different characteristics have been also employed in [8], [9].

2) Feedback control real-time scheduling: There is a number of techniques proposed for feedback control scheduling for distributed real-time systems. These include centralized or decentralized schemes that focus on a) *utilization control* [10],[11], [12], [13], [14], [15], [16] for enforcing desired utilizations of multiple processors in a distributed system, by adapting the rates of end-to-end tasks. Several different control solutions are proposed, ranging from simple localized or distributed regulation controllers, to hybrid, supervisory, hierarchical and optimal controllers, for both continuous and discrete configurations. b) *dynamic resource (task) reallocation* [17] via a combination of local QoS level adaptation (performing admission control and service level ratio adjustments) to achieve deadline miss ratio guarantees.

3) Adaptive control for utility computing: There are many control theoretical frameworks proposed in the literature where feedback controllers are employed to meet QoS goals in utility computing environments.

In such applications, a static resource management policy based on heuristics can lead to several undesirable situations which result in degraded performance and failure to meet QoS goals, specified e.g. by SLA agreements, or to unnecessarily high maintenance and operation cost and corresponding loss of revenue. Such situations include *under-utilization* of the available resources due to over-provisioning, while at the same time *overload* of some of the resources due to sudden workload peaks could also occur. These suggest the need to design an adaptive resource management control system that dynamically adjusts the resource shares in order to meet QoS goals while achieving high resource utilization.

A common solution is the use of a *utilization controller* as a dynamic resource allocator designed to adaptively adjust to varying workloads so that high resource utilization and high-application-level QoS can be achieved. However, under *overload* conditions, performance starts degrading due to resource contention and SLA or QoS goals violations may occur, thus additional controllers are required to employ adaptation mechanisms such as *service quality adjustment (content adaptation)*, *service differentiation* (based on priorities or service ratios) among applications, or even *admission control* that rejects some requests. These controllers can be combined together in hierarchy levels and may employ continuous as well as discrete decision configurations giving rise to nonlinear, switching, hybrid, hierarchical and cascaded schemes, see e.g. [18], [19], [20], [21], [22], [23], [24]. Apparently, the whole architecture could be formulated as an optimization problem expressing trade-offs between the aforementioned conflicting goals, specified by SLAs, such as [25], [26], [27], [28], [29].

C. Key research works

In this section we continue with a more detailed presentation of the most important works.

For the problem of QoS management of a single Web server, the authors in [18] propose a web *content adaptation* scheme based on a server utilization controller that allows content to be adapted under overload conditions, so that both server *underutilization* and serious *resource contention* due to overload can be dealt with. In addition to overload and underutilization protection, the QoS management problem can have extra facets, such as a) *performance isolation and QoS guarantees*, i.e. a maximum delivered bandwidth for a maximum request rate is guaranteed independently for each site, when multiple independent sites are co-hosted on the same machine. This can be done by using virtual servers, b) *service differentiation*, i.e. to support request prioritization, where lower priority customers are degraded first, and c) *excess capacity sharing*, where non-allocated resources on a virtual server are made available to other virtual servers which are overloaded. All these features can be supported by extending the previous architecture.

In [19] the authors propose feedback controllers for the adjustment of entitlement values for a resource container on a server shared by multiple applications. The key question is “what is the minimum amount of system resource an application needs in order to meet its performance objective?”. This problem is solved using a feedback entitlement control, which adjusts entitlement values on the basis of performance measurements. Two controllers are proposed, a fixed PI controller and an adaptive controller. Dynamic models inferred from experimental data using system identification techniques are employed.

The same ideas are continued in [21] where extra experiments reveal that the system’s input-output relation changes with various operating conditions and exhibits a nonlinear bimodal behavior when moving from underload to overload conditions.

The authors extend their previous work in [23] to deal with an enhanced dynamic resource allocation problem using a new *nested* control design. They propose a feedback control

system consisting of two nested integral control loops for managing QoS metrics along with the utilization of the allocated CPU resource. This is an attempt to integrate single loops controlling response time or utilization alone, which were found to be sensitive to the bimodal behavior of the system. Another new ingredient of this work is the consideration (maximization) of the overall utility of the service, which represents a trade-off between better QoS and lower cost of resources.

Previous work is suitable for applications hosted inside a single virtual machine. For multi-tier applications with individual components distributed in different virtual machines, the same authors propose an adaptive *two-layered* controller in [20]. It employs a *utilization* controller that controls the resource allocation for a single allocation tier and an *arbiter* controller that controls the resource allocations across multiple application tiers.

There are also further proposals that go beyond simple control techniques used in the previously mentioned works. More complex techniques are introduced, such as MIMO control, model predictive control with constraints, distributed model predictive control, hybrid supervisory control and hierarchical control. We next highlight those works which are of particular interest in our problem.

In [11] the end-to-end utilization control (EUCON) algorithm is presented where a MIMO model and a Model Predictive Control (MPC) approach are adopted. In [12] they extend the EUCON centralized algorithm to a decentralized (DEUCON) control structure based on recent advances in *distributed model predictive* control theory. This controller can effectively distribute the computation and communication cost to different processors and tolerate considerable communication delay between local controllers. Hence, it provides a scalable and robust utilization control for large-scale distributed real-time systems executing in unpredictable environments.

The previous works propose controllers that rely on the existence of continuous control variables in real-time systems. However, there exist real-time systems that support only a *finite* set of discrete configurations that limit the adaptation mechanisms. The feedback control real-time scheduling algorithms discussed cannot handle effectively discrete control variables, especially when the number of possible values is small. Hybrid (continuous/discrete) control algorithms must be used. Such hybrid control approaches have been developed in [30], [31], [25], [26]. The problem with these works is that they employ exhaustive search algorithms to evaluate a performance measure for all possible operating states during a prediction horizon in order to select the best control input. The exhaustive search introduces significant overhead and is not suitable for real-time systems.

In [15] the authors propose a different approach, the Hybrid Supervisory Utilization Control (HySUCON) algorithm for enforcing utilization bounds in real-time systems by adaptively selecting the task rates from a finite discrete set. In a more recent work [14], the potential drawbacks of the previous strategy for discrete rate adaptation are relaxed by adopting a *Multiparametric* Rate Adaptation (MPRA) algorithm. The key novelty and advantage of MPRA is that it can efficiently produce optimal solutions while reducing the online computation complexity to polynomial time. This is made possible due to offline preprocessing where an NP-hard utility optimization problem (such problems with discrete options can be shown to be NP-hard) is transformed to the evaluation of a piecewise linear function of the CPU utilization. At runtime, MPRA produces optimal solutions by evaluating this function based on the current CPU utilization. This work overcomes the limitations of existing approaches, such as optimal solutions which are computationally expensive and efficient heuristics which are only suboptimal. The MPRA algorithm is based on *multiparametric mixed-integer linear programming (mp-MILP)* which is a general framework for solving mathematical programming problems with constraints that depend on varying parameters.

IV. Discussion

In our context, we have a dynamic resource allocation problem where the goal is to optimize system performance and revenue of the service owner. We assume there is a pool of machines, and there are two control inputs, i.e. we continuously decide on the number of machines allocated to each workload and the implementation (quality) level of each service contained in the workload. On the other hand, for building efficient feedback loops we need to have frequent measurements of important performance (QoS) metrics. These include of course the request *response times* (for which critical deadlines are specified in the SLAs) but are not limited to them. Knowledge of the *utilization* of several resources of interest (CPU, memory, disk or I/O) –especially those that may become bottleneck resources– is also important, since a normally functioning system has to avoid both resource *underutilization* –which in our case may result in reduction of the number of customers that are allowed to connect– and *overload* –which is dangerous and usually results in significant performance degradation–. Underutilization can be avoided by dynamically assigning to each workload or service only the amount of resources which are necessary for its current implementation. In the presence of request bursts, the system may be overloaded, and then our control system has to be able to detect such a situation and respond promptly, by switching to a different mode and using a possibly different urgent adaptation mechanism, such as admission control, i.e. begin rejecting requests.

It is also important to note that our decision variables are basically discrete in nature, i.e. there is a finite set of machines and service implementations. Although this can be relaxed by using continuous variables and employing some kind of mapping technique from continuous values to the discrete set, it is well known that such approximations may be adequate for systems with e.g. a large number of machines, but they can surely become inadequate for systems with a small number of e.g. service levels. Moreover, adaptation mechanisms such as admission control are discrete in nature and SLA specifications are also usually expressed

using piecewise linear step functions. All these suggest that our problem is clearly hybrid (continuous/discrete) in nature, contains several different modes of operation and any optimization solution should carefully take into account transitions and switchings between operating modes, i.e. different models may be used and possibly also different objectives set.

As outlined above, there are several methodologies that can be useful in our context, and can address all the aforementioned issues. We briefly outline the most promising ones in two broad categories.

1) The **indirect** *approach:* Several regulation and/or optimization techniques can be heuristically combined to form a complete control solution to the problem.

The first is *SLA-based heuristic optimization* using a hill-climbing technique, such as *fuzzy* control or *extremum* control. Such optimization could be used when e.g. determining the optimum number of machines allocated to a workload. However, such a controller could be useful only as a *local* controller, i.e. part of a more generalized hierarchical control system.

The second is designing *regulation* controllers for dynamic resource allocation. This is almost always performed by *utilization* controllers. These controllers help avoiding both underutilization and overload. The simplest approach is to employ a simple non-model based (or coarse-model based using a simple linear relation between request rates, throughput and utilization) utilization PI controller for our control inputs and a mapping technique from continuous values to a discrete set of quality levels. Of course, adaptive controllers (with online parameter update) or mixed feedforward-feedback controllers (using queuing models for better prediction) can provide better performance. Again, such controllers cannot solve our problem by themselves, they need coordination (e.g. receive time-varying setpoint commands) from other higher-level controllers etc.

One complete solution is the use of a *supervisory-hierarchical- cascaded* control structure, which may consist of multiple cascaded loops, where the inner ones receive setpoint commands from the outer ones, which at higher levels may also perform logic decision such as mode switching with different objectives and/or adaptation mechanisms. Such approaches have appeared in the literature in several publications, as outlined above. *Hybrid* control ideas can be also quite useful in that respect.

2) The **direct** *approach:* A more elegant and direct approach is to formulate the problem as a constrained optimization problem subject to several constraints and try to solve it using an appropriate algorithm offering a computationally tractable solution for online implementation. For similar problems in the literature, several algorithms have been proposed such as integer programming, dynamic programming, mixed integer-linear programming and multiparametric mixed-integer linear programming (mp-MILP). While all the rest run in exponential time –only sub-optimal approximate solutions run in polynomial time– , we have seen that mp-MILP is the only algorithm that can provide optimal solutions in polynomial time, due to splitting the problem in an off-line component with high complexity and an online with tractable complexity. Hence the use of mp-MILP solvers seems a very good solution for our problem.

Furthermore, another interesting perspective is to formulate the problem as an *optimal control* problem and use modern design methodologies and numerical tools for its complete solution. In this respect, a very promising path is to formulate our problem in a *model predictive control* (MPC) context and the *Mixed Logical Dynamical* (MLD) paradigm [32], [33] and solve it using powerful multi-parametric solvers which are widely available (e.g. the multi-parametric toolbox for MATLAB [34]). This more general setting can be based on both linear and quadratic performance criteria, include state-space system models to capture dynamics and predict transient behavior, include logical variables that represent different modes of operation and also exploit several advantages of MPC as a control algorithm for constrained control of hybrid systems.

Finally, recent research efforts suggesting co-design of the control with optimal real-time scheduling [35] have shown that better control performance can be obtained which allows more space for the implementation of control algorithms with higher computational complexity.

V. Conclusions

Application of control theory concepts and algorithms to computing systems is not new. The autonomic computing vision has benefitted from the use of feedback control theory as evidenced by the large number of relevant publications, both for small scale centralized systems (see e.g. [36], [37] and references therein), and for large scale distributed extensions [38].

Motivated by the recent booming in the newly-founded area of cloud computing, in this paper we review the most influential research efforts in the literature, where control theory has been employed for solving related problems of QoS guarantees and utility computing. Our survey suggests that cloud computing and the associated large scale resource management and performance/revenue maximization problems could be effectively tackled by turning to advanced control methodologies such as supervisory, cascaded, hybrid and optimal control. We believe that sophisticated control theoretical solutions, characterized by a solid theoretical foundation and performance guarantees, and also accompanied by reliable algorithms and well-tested numerical tools, have strong potential to overcome heuristics in performance. Future work will proceed with a thorough investigation of these suggestions in simulation studies and real empirical evaluation experiments.

In this work, our focus has been on the resource allocation problem exclusively. However, if the cloud resources are interconnected by a slow and unreliable network, then any solution should address a number of additional issues, including the problems of network instrumentation and security.

VI. Acknowledgements

C.A. Yfoulis has been supported by the ATEI grant 477/4-7-2007 titled “Adaptive QoS control and optimization of computing systems”.

References

[1] L. Wang, J. Tao, M. Kunze, A. C. Castellanos, D. Kramer, and W. Karl, “Scientific cloud computing: Early definition and experience,” *High Performance Computing and Communications, 10th IEEE International Conference on*, pp. 825–830, 2008.

[2] M. A. Vouk, “Cloud computing - Issues, Research and Implementations,” in *Int. Conf. on Information technology Interfaces*, 2008.

[3] Z. Liu, M. S. Squillante, and J. L. Wolf, “On maximizing service-level-agreement profits,” in *EC ’01: Proceedings of the 3rd ACM conference on Electronic Commerce*, 2001, pp. 213–223.

[4] A. Chandra, W. Gong, and P. J. Shenoy, “Dynamic Resource Allocation for Shared Data Centers Using Online Measurements,” in *IWQoS*, 2003, pp. 381–400.

[5] J. L. Hellerstein, K. Katircioglu, and M. Surendra, “An on-line, business-oriented optimization of performance and availability for utility computing,” *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 10, pp. 2013–2021, 2005.

[6] D. A. Menascé, D. Barbará, and R. Dodge, “Preserving QoS of e-commerce sites through self-tuning: a performance model approach,” in *ACM Conference on Electronic Commerce*, 2001, pp. 224–234.

[7] Y. Diao, J. L. Hellerstein, and S. S. Parekh, “Using fuzzy control to maximize profits in service level management,” *IBM Systems Journal*, vol. 41, no. 3, pp. 403–420, 2002.

[8] A. Gounaris, C. Yfoulis, R. Sakellariou, and M. D. Dikaiakos, “A control theoretical approach to self-optimizing block transfer in web service grids,” *ACM Transactions on Autonomous and Adaptive Systems*, vol. 3, no. 2, 2008.

[9] ——, “Robust Runtime Optimization of Data Transfer in queries over Web Services,” in *ICDE*. IEEE, 2008, pp. 596–605.

[10] C. Lu, J. A. Stankovic, and S. H. Son, “Feedback control real-time scheduling: Framework, modeling and algorithms,” *Journal of Real-Time Systems, Special Issue on Control-Theoretical Approaches to Real-Time Computing*, vol. 23, pp. 85–126, 2002.

[11] C. Lu, X. Wang, and X. D. Koutsoukos, “Feedback Utilization Control in Distributed Real-Time Systems with End-to-End Tasks,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 16, no. 6, pp. 550–561, 2005.

[12] X. Wang, D. Jia, C. Lu, and X. D. Koutsoukos, “DEUCON: Decentralized End-to-End Utilization Control for Distributed Real-Time Systems,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 18, no. 7, pp. 996–1009, 2007.

[13] N. Shankaran, X. D. Koutsoukos, D. C. Schmidt, Y. Xue, and C. Lu, “Hierarchical control of multiple resources in distributed real-time and embedded systems,” *Real-Time Systems*, vol. 39, no. 1-3, pp. 237–282, 2008.

[14] Y. Chen, C. Lu, and X. D. Koutsoukos, “Optimal Discrete Rate Adaptation for Distributed Real-Time Systems,” in *RTSS*. IEEE Computer Society, 2007, pp. 181–192.

[15] X. D. Koutsoukos, R. Tekumalla, B. Natarajan, and C. Lu, “Hybrid Supervisory Utilization Control of Real-Time Systems,” in *IEEE Real-Time and Embedded Technology and Applications Symposium*. IEEE Computer Society, 2005, pp. 12–21.

[16] Y. Fu, H. Wang, C. Lu, and R. S. Chandra, “Distributed Utilization Control for Real-Time Clusters with Load Balancing,” in *RTSS*. IEEE Computer Society, 2006, pp. 137–146.

[17] J. A. Stankovic, T. He, T. F. Abdelzaher, M. Marley, G. Tao, S. H. Son, and C. L, “Feedback Control Scheduling in Distributed Real-Time Systems,” in *IEEE Real-Time Systems Symposium*. IEEE Computer Society, 2001, pp. 59–.

[18] T. F. Abdelzaher and N. Bhatti, “Web Server QoS Management by Adaptive Content Delivery,” in *International Workshop on Quality of Service*, 1999.

[19] X. Liu, X. Zhu, S. Singhal, and M. F. Arlitt, “Adaptive entitlement control of resource containers on shared servers,” in *Integrated Network Management*, 2005, pp. 163–176.

[20] 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 *EuroSys*, P. Ferreira, T. R. Gross, and L. Veiga, Eds. ACM, 2007, pp. 289–302.

[21] Z. Wang, X. Zhu, and S. Singhal, “Utilization and SLO-Based Control for Dynamic Sizing of Resource Partitions,” in *DSOM*, ser. Lecture Notes in Computer Science, J. Schönwälder and J. Serrat, Eds., vol. 3775. Springer, 2005, pp. 133–144.

[22] C. A. Santos, A. Sahai, X. Zhu, D. B. 0002, V. Machiraju, and S. Singhal, “Policy-Based Resource Assignment in Utility Computing Environments,” in *DSOM*, ser. Lecture Notes in Computer Science, A. Sahai and F. Wu, Eds., vol. 3278. Springer, 2004, pp. 100–111.

[23] X. Zhu, Z. Wang, and S. Singhal, “Utility-Driven Workload management using Nested Control Design,” in *American Control Conference*, 2006, pp. 6033–6038.

[24] A. Chandra, W. Gong, and P. J. Shenoy, “Dynamic resource allocation for shared data centers using online measurements,” in *IWQoS*, ser. Lecture Notes in Computer Science, K. Jeffay, I. Stoica, and K. Wehrle, Eds., vol. 2707. Springer, 2003, pp. 381–400.

[25] N. Kandasamy, S. Abdelwahed, G. C. Sharp, and J. P. Hayes, “An Online Control Framework for Designing Self-Optimizing Computing Systems: Application to Power Management,” in *Self-star Properties in Complex Information Systems*, ser. Lecture Notes in Computer Science, Ö. Babaoglu, M. Jelasity, A. Montresor, C. Fetzer, S. Leonardi, A. P. A. van Moorsel, and M. van Steen, Eds., vol. 3460. Springer, 2005, pp. 174–188.

[26] M. Khandekar, N. Kandasamy, S. Abdelwahed, and G. C. Sharp, “An online predictive control framework for designing self-managing computing systems,” *Multiagent and Grid Systems*, vol. 1, no. 2, pp. 63–72, 2005.

[27] S. Abdelwahed, N. Kandasamy, and S. Neema, “A control-based framework for self-managing distributed computing systems,” in *WOSS*, D. Garlan, J. Kramer, and A. L. Wolf, Eds. ACM, 2004, pp. 3–7.

[28] S. Abdelwahed, S. Neema, J. P. Loyall, and R. Shapiro, “A Hybrid Control Design for QoS Management,” in *RTSS*. IEEE Computer Society, 2003, pp. 366–370.

[29] S. Shelford, M. M. Akbar, E. G. Manning, and G. C. Shoja, “Distributed Optimal Admission Controllers for Service Level Agreements in Interconnected Networks,” in *Applied Informatics*, M. H. Hamza, Ed. IASTED/ACTA Press, 2003, pp. 565–570.

[30] S. Abdelwahed, N. Kandasamy, and S. Neema, “Online Control for Self-Management in Computing Systems,” in *IEEE Real-Time and Embedded Technology and Applications Symposium*. IEEE Computer Society, 2004, pp. 368–375.

[31] N. Kandasamy, S. Abdelwahed, and J. P. Hayes, “Self-Optimization in Computer Systems via On-Line Control: Application to Power Management,” in *ICAC*. IEEE Computer Society, 2004, pp. 54–61.

[32] A. Bemporad, M. Morari, V. Dua, and E. Pistikopoulos, “The Explicit Solution of Model Predictive Control via Multiparametric Quadratic Programming,” in *American Control Conference*, Chicago, USA, 2000, pp. 872–876.

[33] A. Bemporad and M. Morari, “Control of systems integrating logic, dynamics, and constraints,” *Automatica*, vol. 35, no. 3, pp. 407–427, 1999.

[34] M. Kvasnica, P. Grieder, and M. Baotić, “Multi-Parametric Toolbox (MPT),” 2004. [Online]. Available: http://control.ee.ethz.ch/ mpt/

[35] M.-M. Ben Gaid, A. Çela, Y. Hamam, and C. Ionete, “Optimal scheduling of control tasks with state feedback resource allocation,” in *In Proceedings of the 2006 American Control Conference*, Minneapolis, Minnesota, USA, June 2006.

[36] J. Hellerstein, D. Tilbury, Y. Diao, and S. Parekh, *Feedback Control of Computing Systems*. Wiley, 2004.

[37] Y. Diao, J. L. Hellerstein, S. S. Parekh, R. Griffith, G. E. Kaiser, and D. B. Phung, “A control theory foundation for self-managing computing systems,” *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 12, pp. 2213–2222, 2005.

[38] Y. Diao, J. L. Hellerstein, and S. Parekh, “Control of large scale computing systems,” *SIGBED Rev.*, vol. 3, no. 2, pp. 17–22, 2006.