Dynamic Trade-off Analysis of QoS and Energy Saving in Admission Control for Web Service Systems

C. Poussot Vassal*
Politecnico di Milano
32, Piazza L. da Vinci
20133 Milano, Italy
charles.poussot@gmail.com

M. Tanelli†
Politecnico di Milano
32, Piazza L. da Vinci
20133 Milano, Italy
tanelli@elet.polimi.it

M. Lovera
Politecnico di Milano
32, Piazza L. da Vinci
20133 Milano, Italy
lovera@elet.polimi.it

Keywords

Performance evaluation, Admission Control, Dynamic Voltage Scaling, Linear Parameter Varying Models, Model Predictive Control.

ABSTRACT

The complexity of Information Technology (IT) systems is steadily increasing. System complexity has been recognised as the main obstacle to further advance of IT and has recently raised energy management issues. Control techniques have been proposed and successfully applied to design Autonomic Computing systems, i.e., systems able to manage themselves trading-off system performance with energy saving goals. As users behaviour is highly time varying and workload conditions can change substantially within the same business day, the Linear Parametrically Varying (LPV) framework seems very promising for modeling such systems. In this paper, a control theoretic method to investigate the trade-off between Quality of Service (QoS) requirements and energy saving objectives in the case of admission control in Web service systems is proposed. First, a dynamic model of the admission control dynamics is estimated via LPV identification techniques. Secondly, an optimisation problem within the Model Predictive Control (MPC) framework is setup, based on the estimated LPV model, by means of which it is possible to investigate the optimal trade-off policy to manage QoS and energy saving objectives.

1. INTRODUCTION

The steady increase in the complexity of Information Technology (IT) systems led IBM to release, in mid-October 2001, the "Autonomic Computing Manifesto" [1] observing that current applications have reached the size of several millions of lines of code while physical infrastructures include thousands of heterogeneous servers and require skilled IT professionals to install, configure, tune, and maintain. System complexity has been recognised as the main obstacle to further advance of IT.

Another emerging problem in this context is related to energy management. The growth in the number of servers has caused an enormous spike in electricity usage. IT analysts predict that, by 2012, up to 40% of an enterprise technology budget will be consumed by energy costs. From an environmental point of view, overall, IT accounts for 2% of global CO_2 emissions, i.e., IT pollutes to the same extent as the global air traffic [2].

Furthermore, system operation has to cope with the variability of users' behaviour and application workloads. Nowadays, IT systems have to provide to their users prescribed Quality of Service (QoS) levels usually defined in terms of application performance, such as requests response time or system throughput. QoS requirements are difficult to satisfy, since workload may vary by several orders of magnitude within the same business day [3]. To handle workload variations and meet QoS requirements, resources have to be dynamically allocated among running applications and the IT architecture has to be re-configured at run-time.

IBM has proposed the Autonomic Computing paradigm as a solution to IT complexity, energy consumption, and run-time reconfiguration issues [1]. The basic idea is to allow IT systems to manage themselves, as the human autonomic nervous system governs basic body functionalities such as heart rate or body temperature, thus freeing the conscious brain – IT administrators – from the burden of dealing with low-level vital functions.

Control theoretic techniques have been proposed and successfully applied to the design of Autonomic Computing systems [4,5]. The main actuation mechanisms which have been implemented are: (i) *Dynamic Voltage Scaling* of server CPUs, (ii) *admission control*, and (iii) *resource allocation in virtualised environments*. This paper will focus on the first two of them.

Dynamic Voltage Scaling (DVS) is a mechanism which can be exploited to reduce a server energy consumption [6–8]. Modern CPUs allow varying both the CPU supply voltage and operating frequency. The adoption of DVS as a control variable is very promising, as power consumption is proportional to the cube of the operating frequency, while server performance varies linearly with the operating frequency. Hence, under light load conditions energy consumption can be effectively reduced by lowering CPU frequency without worsening the provided QoS level.

Admission control is an overload protection mechanism which rejects requests under peak workload conditions in order to provide performance guarantees to the running applications [9–11]. Admission control is effective if the admitted requests are served according to their QoS constraints. However, QoS constraints need to be traded-off with energy saving objectives. To this end, it is

*C. Poussot Vassal is also with ONERA/DCSD, BP 74025, 31055 Toulouse cedex, FRANCE.

†The work of M. Tanelli is partially supported by the Green Active Management of IT Systems (GAME-IT) project, funded by the Politecnico di Milano.

ROSSA 2009, October 19, Pisa, Italy

DOI 10.4108/ICST.VALUETOOLS2009.7941

of great importance to be able of establishing the optimal trade-off policy at *design time*. This paper presents a control oriented methodology to handle such issue.

In the utility-based context developed within the autonomic computing framework, approaches have been introduced to analyse and optimize the degree of users' satisfaction, expressed in terms of user-level performance metrics, which typically use performance models based on queueing theory [12–14]. These approaches can handle multiple decision variables (e.g., joint admission control and resource allocation [15]) but rely on the assumption that the system is at *steady state*. Hence, the optimal trade-off management policies that such approaches convey do not take into account the dynamic nature of the underlying physical system, which can have a decisive importance in selecting the management policies at machine level. As a matter of fact, in the formulation of autonomic computing as a control problem, the most critical issue is the variability of the dynamics of the Web server as a function of workload.

To capture such a variability and take it into account in the trade-off analysis, we propose to model the Web service dynamics via Linear Parametrically Varying (LPV) models, which have proved effective – in the control community – to deal with such parameter dependent systems, see e.g., [16, 17]. Recent work has shown that the LPV framework is very general, since it allows describing the performance of an IT system by exploiting all of the available technological mechanisms to manage QoS [18, 19]. Thus, in this work we first perform identification and validation experiments aimed at estimating an LPV dynamic model of the admission control dynamics. Such a model constitutes the basis for formulating the optimisation problem, the solution of which provides the optimal policy to manage the QoS/Energy trade-off. Specifically, the problem is solved within the Model Predictive Control (MPC) framework, which is a widely used control approach allowing to formulate the control problem as a constrained optimisation one, see e.g., [20,21]. Thus, the performance evaluation problem associated with the Web service system under study will be formulated in terms of a suitable cost function expressing the QoS/Energy saving trade-off and of a number of constraints further specifying the control objectives.

Notably, the proposed approach is quite general, in that it can accommodate different QoS measures together with other energy-related constraints within the same framework, and, most importantly, it can be applied to all available technological mechanisms to manage QoS with no conceptual differences. Another advantage is that the *ideal* formulation of the MPC control problem used for performance evaluation can be easily adapted to obtain a controller which can be actually run on the real system, so as to provide the possibility of analyzing and quantifying the discrepancy between the optimal trade-off policy and the performance which can be achieved on the real system at *design time*.

The structure of the paper is as follows. Section 2 provides the necessary background and defines the notation for the problem under study. Section 3 illustrates the LPV state space models employed for identification and shows the approach used to identify reliable models of the admission control dynamics, together with the experimental setting employed for identification and validation experiments. In Section 4 the proposed MPC-based approach to the analysis of the QoS/Energy trade-off is illustrated, and simulations results are shown to assess its suitability for the considered application.

2. PROBLEM STATEMENT AND NOTATION

In the following, a Web service system which provides DVS and admission control will be considered. For the sake of simplicity, Web service applications are configured to serve requests according to the FIFO policy and run on a single CPU. In the queueing theory context, [22], the following quantities are commonly employed to describe the incoming workload over a time interval $[k\Delta t,\ (k+1)\Delta t]$, where Δt is the sampling interval:

- λ_k denotes the average requests arrival rate for the Web service application in the k-th time interval;
- s_k is the average requests service time, i.e., the overall CPU time needed to process a request for the considered application in the k-th time interval when the CPU runs at the maximum frequency and the CPU is fully dedicated to the execution of the application;
- T_k is the average server response time, i.e., the overall time a request stays in the system in the k-th time interval;
- X_k denotes the average Web service throughput, defined as the number of served requests in the k-th time interval;
- f_k is the ratio between the frequency adopted by the server CPU in the time interval k with respect to the maximum CPU frequency;
- $\mathscr{P}_k$ is the probability that a request will be admitted to the system in the k-th time interval.

The service time represents the overall server CPU time needed to serve a customer. Note that, generally speaking, the service time can be regarded as inversely proportional to the server CPU frequency. When physical servers are endowed with DVS capabilities, in fact, the effect of - say - lowering the CPU frequency when a light workload is present in the system causes an increase of the effective CPU time needed to serve a request (see [7]). This assumption is supported by current technology trends, since in modern systems (e.g., AMD Opteron 2347HE Barcelona core) CPUs and RAM clock can be scaled independently. Thus, in what follows it will be assumed that the effective service time can be defined as $s_{f,k} = s_k/f_k$. The inverse of such quantity is commonly referred to as the maximum service rate and indicated with $\mu_k = 1/s_{f,k}$. This assumption has been validated experimentally on a Web server endowed with the voltage scaling capability; the results of the validation experiment are depicted in Figure 1. As can be seen, the above relationship is accurate (with an approximation of about 2%) and therefore can be considered acceptable for the present purposes.

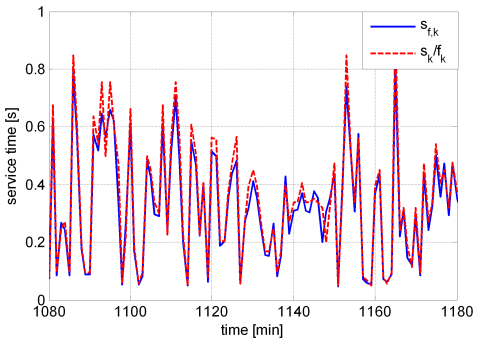

Figure 1: Experimental verification of $s_{f,k} = s_k/f_k$ on a Web server with DVS functionalities.

The effect of admission control is to reduce the number of requests served by the system. More precisely, the requests throughput X_k is related to the requests arrival rate λ_k and the probability of admission $\mathscr{P}_k$ by $X_k = \mathscr{P}_k \lambda_k$.

Digital Object Identifier: 10.4108/ICST.VALUETOOLS2009.7941
http://dx.doi.org/10.4108/ICST.VALUETOOLS2009.7941

As already mentioned, classical queueing theory provides a description of the system which relies on steady-state assumptions, and it is therefore reliable only over long time horizons. In this context, considering for the sake of simplicity a single application, a queueing system is said to be *stable* if $X_k < \mu_k$, which is the condition that guarantees that the queue does not overflow due to an overly low effective service time or too large throughput. This assumption is only considered for simplicity, even if the queueing system could in principle be stable over multiple time intervals with $X_k \geq \mu_k$ for some of the time intervals, as long as the traffic intensity of other time intervals is sufficiently small to balance the local overload phenomena. Note, in fact, that we seek a dynamic model for the server behaviour, which does not rely on queueing theory assumptions.

By means of Little's law, [22], queueing theory predicts that the steady-state average response time for a stable open system (that is for a system in which the number $\mathcal{N}$ of requests in the queue is variable), can be computed as

$$\overline{T} = \frac{\overline{\mathcal{N}}}{\overline{X}}. \tag{1}$$

For control purposes, however, a dynamical model of the server capable of capturing transients must be derived. We recall, in fact, that one of the aims of the present work is to obtain a control-oriented dynamical description of the server behaviour to be employed for dynamic performance evaluation via MPC control.

REMARK 1. *As incoming requests – independently of the application they may try to access – wait in a queue before accessing the physical server, it is clear that the system dynamics will have a feedthrough term, i.e., a direct path from input to output. In fact, the response time T_k (i.e., the system output) is given by*

$$T_k = s_{f,k} + \xi_k, \tag{2}$$

where the queueing time ξ_k accounts for the time that the request spends in the queue. As such, the real dynamics of the system are to be found in the variable $\xi_k = T_k - s_{f,k}$. In what follows, LPV models will be identified for the dynamics of ξ_k only, and the final response time will then be retrieved via Equation (2)*.*

3. LPV STATE SPACE MODELS: IDENTIFICATION AND VALIDATION

In the considered application, an LPV modeling formulation has been adopted to handle the system nonlinearities due to workload variations. As shown in e.g., [18], this choice is essential since a simple Linear Time Invariant (LTI) model would not be precise enough to capture all the relevant dynamic behaviour of the considered system. In this Section, the LPV system structure is described, and the identification approach used to model the dynamics of the admission control system are briefly presented (for more details on the identification approach, refer to [18, 19]). Further, the experimental setup employed for the identification and validation experiments is detailed, together with the obtained results.

3.1 LPV Model Identification Approach

LPV systems are linear time-varying plants whose state space matrices are fixed functions of some vector of measurable, time varying parameters. LPV model identification algorithms are available in the literature both for input/output and state space representations of parametrically-varying dynamics. In particular, in the recent works [6, 23] an input-output modelling approach was adopted. If, however, the aim of the identification procedure is to eventually work out LPV models in state space form for control design purposes, one should keep in mind that the usual equivalence notions applicable to LTI systems cannot be directly used in converting LPV models from input-output to state space form, as the *time-variability* of LPV systems ought to be taken into account (see, e.g., the discussion in [24]). Bearing this in mind, in this work we focus on state space LPV models in the form

$$\begin{array}{rcl} x_{k+1} & = & A(p_k)x_k + B(p_k)u_k \\ y_k & = & C(p_k)x_k + D(p_k)u_k, \end{array} \tag{3}$$

where $p \in \mathbb{R}^s$ is the parameter vector and $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$, $y \in \mathbb{R}^l$. It is often necessary to introduce additional assumptions regarding the way in which p_k enters the system matrices: in this work we focus on affine and input-affine models, defined as follows.

1. Affine parameter dependence (LPV-A):

$$A(p_k) = A_0 + A_1 p_{1,k} + \ldots + A_s p_{s,k} \tag{4}$$

   and similarly for B, C and D, and where by $p_{i,k}, i = 1, \ldots, s$ we denote the i-th component of vector p_k. This form can be immediately generalised to polynomial parameter dependence.

2. Input-affine parameter dependence (LPV-IA): this is a particular case of the LPV-A parameter dependence in which only the B and D matrices are considered as parametrically-varying, while A and C are assumed to be constant, i.e., $A = A_0$, $C = C_0$.

As far as LPV model identification is concerned, it is usually convenient to consider first the simplest form, i.e., the LPV-IA one, as its parameters can be retrieved using Subspace Model Identification (SMI) algorithms for LTI systems by suitably extending the input vector. In this work the MOESP class of SMI algorithms (see [25]) has been considered. LPV-IA models also provide a useful initial guess for iterative methods which can be used for the identification of fully parameterised models in LPV-A form, along the lines of [26, 27]. The classical way to perform linear system identification is by minimizing the error between the real output and the predicted output of the model. A similar approach can be used for LPV state-space systems of the form (3). Letting the system matrices of (3) be completely described by a set of parameters θ, identification can be carried out by minimizing the cost function

$$V_N(\theta) := \sum_{k=1}^{N} ||y_k - \hat{y}_k(\theta)||_2^2 = E_N^T(\theta) E_N(\theta),$$

with respect to θ, where

$$E_N^T(\theta) = \left[\left(y_1 - \hat{y}_1(\theta)\right)^T \quad \cdots \quad \left(y_N - \hat{y}_N(\theta)\right)^T \right],$$

y_k denotes the measured output and $\hat{y}_k(\theta)$ denotes the output of the LPV model to be identified and $||v||_2$ indicates the 2-norm of vector v.

3.2 Testbed setting and experimental results

To perform the experiments needed for collecting identification and validation data, a workload generator and a micro-benchmarking Web service application have been used. The workload generator is based on a custom extension of the Apache *JMeter* 2.3.1 workload injector, [28], which allows to generate workload according to an open model [22] with a Poisson arrival process. As a matter of fact, to approximate the behaviour of internet requests, the workload injector generates traffic according to a Poisson distribution. This is

Digital Object Identifier: 10.4108/ICST.VALUETOOLS2009.7941
http://dx.doi.org/10.4108/ICST.VALUETOOLS2009.7941

motivated by the fact that several analyses of actual e-commerce site traces have shown that the Internet workload follows a Poisson distribution with a good approximation, [29]. The analysis of burstiness behaviour and long range dependent phenomena is left as part of future work, [30].

The micro-benchmarking Web service application is implemented as a Java servlet and is hosted within the Apache *Tomcat* 6.0 application server, [31]. The servlet has been designed to consume a fixed CPU time which allows to emulate the DVS of the physical server (a Pentium D machine with no DVS support). The benchmarking code has been embedded within the *synchronised* Java construct in order to schedule requests execution according to a FIFO policy. The application has been instrumented to accurately determine the service time of each request; note, however, that this is not a limitation as there exist several techniques to assess the number of CPU cycles consumed by requests both at application level (e.g., the Application Resource Measurement API, [32]) or at operating system level (e.g., kernel-based measurements, [11]).

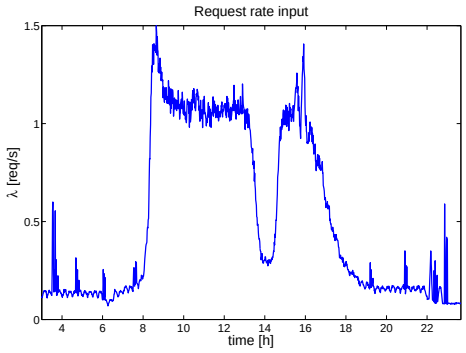

Figure 2: Time history of the request rate λ_k applied for LPV model identification and validation.

For admission control dynamic modeling, the identification experiments have been carried out employing the synthetic workload profile discussed above. Specifically, the request rate behaviour is the one shown in Figure 2, and the application service time follows a log-normal distribution with $\sigma[s_k] = 4E[s_k]$.

As for the admission probability $\mathscr{P}_k$, it varies stepwise every 1 minute, with values between 0.1 and 1. Figure 3 shows the time history of the resulting server utilization (which for the case of admission control can be defined as $\rho_{ac,k} = \lambda_k s_k \mathscr{P}_k$) employed in the identification experiments.

For validation purposes, different realisations of the same workload profile have been employed, while varying the parameters of the service time log-normal distribution. Specifically, two validation tests have been performed with the standard deviation of the log-normal $\sigma[s_k] = qE[s_k]$ and $q = \{2,6\}$. This allows to analyse whether the identified models are sensitive to the variability of the CPU time distribution.

To quantitatively evaluate the models, both on identification and validation data, two metrics will be considered: the percentage Variance Accounted For (VAF), defined as

$$VAF = 100\left(1 - \frac{Var[y_k - y_{sim,k}]}{Var[y_k]}\right), \tag{5}$$

where y_k is the measured signal and $y_{sim,k}$ is the output obtained by simulating the identified model, and the percentage average error

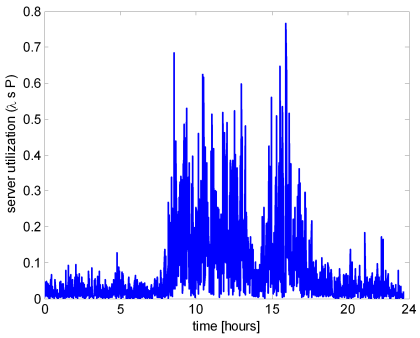

Figure 3: Time history of the server utilization $\rho_{ac,k} = \lambda_k s_k \mathscr{P}_k$ used in the identification experiments for admission control.

e_{avg}, computed as

$$e_{avg} = 100\left|\frac{E_t[y_k - y_{sim,k}]}{E_t[y_k]}\right|. \tag{6}$$

This choice allows to assess the system performance both in a 1 and in a 2-norm sense.

Based on the identification data, an LPV-IA model for the admission control dynamics was estimated. Specifically, we consider as input the admission probability, i.e., $u_k = \mathscr{P}_k$ and use the service time s_k, the application utilization $\rho_k = s_k\lambda_k$ and its square as scheduling parameters, that is $p_k = [s_k \;\; \rho_k \;\; (\rho_k)^2]$. The system output is the server response time T_k. The model order was set to 2 via cross-validation analysis and the best sampling time for this application proved to be $\Delta t = 1$ min.

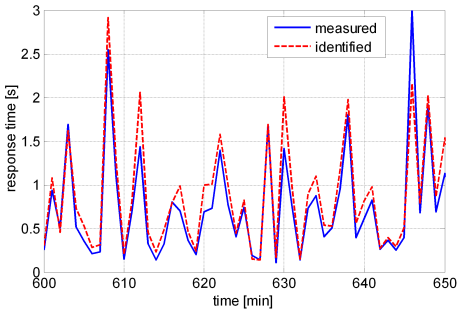

Figure 4: Detail of the measured (solid line) and simulated (dashed line) response time obtained with an LPV-IA model for the admission control dynamics on validation data.

Based on the discussion in Section 2, an LPV-IA model for the dynamics of ξ_k was estimated, and the overall response times computed according to equation (2). A plot of the simulated response time obtained with the identified model on validation data is compared to the measured one in Figure 4 for the case of $q = 2$. For the considered data (see also Figure 3), the light load data are those in the time interval $t \in [0,9) \cup (20,24]$ h, while the heavy load data in the time interval $t \in [9,20]$h.

A summary of the identified model performance on validation data for both $q = 2$ and $q = 6$ is provided in Table 1, which confirms the validity of the proposed LPV model.

Digital Object Identifier: 10.4108/ICST.VALUETOOLS2009.7941
http://dx.doi.org/10.4108/ICST.VALUETOOLS2009.7941

Valid. Performance $\Delta t = 1$ min	$q = 2$	$q = 6$
VAF on 24h	78.38%	74.96%
VAF light load	92.63%	83.01%
VAF heavy load	73.79%	63.57%
e_{avg} on 24h	3.35%	6.60%
e_{avg} light load	0.42%	2.85%
e_{avg} heavy load	5.48%	10.74%

Table 1: Performance of the identified models for admission control with $\Delta t = 1$ min on validation data.

4. DYNAMIC ANALYSIS OF THE QOS/ENERGY TRADE-OFF

The main challenge related with QoS requirements can be stated as that of guaranteeing a given service time associated to a request, denoted with T_{ref}, which is defined and negotiated between the customers and the service provider, and a certain minimal amount of accepted requests (here modeled as an admission probability $\mathscr{P}_k$), while either:

1. Maximizing the number of served users: Quality of Service (QoS) objective.

2. Minimizing the power consumption: Energy saving objective.

These objectives reflect the natural QoS/Energy saving trade-off. Indeed, the first objective reflects the ability of a service provider to serve a large number of requests, while the second one reflects the ability of a service provider to achieve energy saving, and thus costs reduction.

In this Section, the optimal performance level of the server is computed using an optimisation algorithm inspired by the Linear Parameter Varying - Model Predictive Control (LPV-MPC) approach.

To this purpose, the relevant background on MPC is first provided in Section 4.1, while the application of MPC to optimal performance computation is presented in Section 4.2.

4.1 Model Predictive Control for Performance Evaluation

Model Predictive Control (MPC) is a widely used approach to the solution of large scale, multivariable, possibly constrained control problems which has been developed in the Control community over the last three decades. The main idea of MPC can be summarised as follows: i) the control problem is formulated as an optimisation one, based on a mathematical model for the plant (and possibly of known external disturbances), a cost function expressing the desired performance of the system over a future time horizon and all the relevant constraints on the input, state and output variables; ii) the control action over the future horizon is computed by repeatedly solving the optimisation problem on line; iii) the implementation of the computed control action is based on the so-called *receding horizon* principle, i.e., at each time step only the first sample of the computed control sequence is actually applied and the control problem is re-solved at the subsequent time step, [20, 21].

MPC is a very attractive idea, as it allows a very natural formulation of control problems in terms of constrained optimisation. On the other hand, however, this approach leads to a number of issues when it comes to guaranteeing closed-loop stability, dealing with uncertainty in the mathematical model of the plant and ensuring that the on-line optimisation problem is computationally tractable in view of the need to solve it on line. Such issues are a subject of active research and have been successfully clarified in a number of frameworks, see e.g., [33–35]. Most of the attention in the literature focused on *standard* MPC problems in which the system under study is LTI and the cost function expressing the desired closed-loop performance is a quadratic one. In this respect, the literature on LPV-MPC is relatively recent, see e.g., [36–38].

When it comes to performance evaluation, MPC is a very valuable tool for a number of reasons. First of all, provided that the system for which a performance analysis is sought after falls within the assumptions under which MPC can be applied with some guarantees, an MPC controller can be set up and implemented fairly easily, as it requires little tuning with respect to other approaches. Furthermore, if, e.g., one is interested in assessing *optimal* performance in the face of external disturbances, the controller can be implemented by assuming that the future evolution of the disturbances is exactly known over the future horizon. Such setting provides the ability of analysing the optimal performance achievable on the considered system, but can be also easily adapted to provide a controller which can be actually implemented in the non ideal case. As an example, in the considered application, to translate the *ideal* LPV-MPC controller used for performance evaluation into a regulator which can be used on the real Web server it is sufficient to design a state estimator from input/output data – the MPC controller needs to have access to the state variables of the dynamical model which are often not measurable – and to let the controller work with no preview of the future evolution of the disturbances (or, possibly, with the evolution estimated by workload predictors, [39]). Of course, the resulting MPC controller performance will be fully and easily comparable with that provided by the ideal case used for performance evaluation, so that the degree of *sub-optimality* of the final implementation can be also investigated at design time.

In the following Section the performance evaluation problem associated with the Web service system under study will be formulated in terms of a suitable cost function expressing the QoS/Energy saving trade-off and of a number of constraints further specifying the control objectives. As a mathematical model for the plant, the identified LPV model described in Section 3 will be used.

4.2 Optimal Performance Analysis

Let us consider the admission probability $\mathscr{P}_k$ and the effective service time $s_{f,k}$ as the control variables. The objective is to compute the theoretical optimal performance of a controlled server. To do so, we work under the following assumptions:

1. The request rate λ_k and the requests service time s_k are considered as known variables over a given time horizon N.

2. The state variables x_k of the system are assumed to accessible.

3. The LPV system model of the Web server dynamics is assumed to be perfectly fitting the real Web server system.

4. The desired response time T_{ref} is considered as a known variable over a given time horizon N.

Based on these assumptions, the idea is to minimise an appropriate performance index representing either the QoS or the Energy saving objective while guaranteeing that the constraints on the system dynamics, control variables and system performance are fulfilled. Note that, as discussed in the previous Section, assumptions 1), 2) and 3) are only valid in an ideal setting, whereas assumption 4) is always fulfilled in practice.

To apply the proposed LPV-MPC approach, one has to define the following:

Digital Object Identifier: 10.4108/ICST.VALUETOOLS2009.7941
http://dx.doi.org/10.4108/ICST.VALUETOOLS2009.7941

1. A cost function to be minimised, representing the trade-off between the QoS and Energy saving objectives, see Section 4.3.

2. A set of dynamic equality constraints, denoted as $\Sigma(x,\lambda,\mathscr{P},s_f)$, based on the LPV system model described in Section 3, see Section 4.4.

3. A set of inequality constraints, denoted as Λ, aimed at guaranteeing that the control signals evolve within physical bounds (input constraints) and that the tracking of the desired response time T_{ref} is achieved (performance constraints), see Section 4.5.

Once such quantities have been defined, the solution to the problem is obtained by solving a nonlinear constrained optimisation problem at each sampling step. The general scheme of the proposed approach is illustrated in Figure 5.

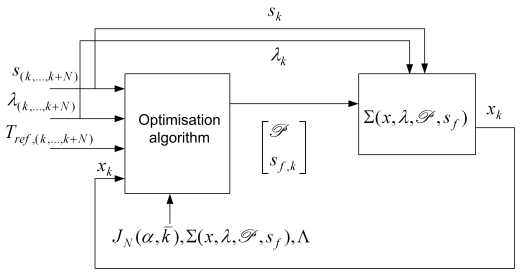

Figure 5: Optimal performance computation scheme.

4.3 Cost function definitions

To model the QoS/Energy saving trade-off, the following cost function can be introduced

$$J_N(\alpha,\bar{k}) = \alpha J_{QoS}(\bar{k}) + (1-\alpha)J_{ES}(\bar{k}) = \\ = \alpha \sum_{k=\bar{k}}^{\bar{k}+N-1} \left| \frac{\overline{\mathscr{P}} - \mathscr{P}_k}{\overline{\mathscr{P}} - \underline{\mathscr{P}}} \right| + (1-\alpha) \sum_{k=\bar{k}}^{\bar{k}+N-1} \left| \frac{\overline{s}_f - s_{f,k}}{\overline{s}_f - \underline{s}_f} \right|, \quad (7)$$

where $k \in \mathbb{N}$ is the current time instant, $\bar{k}$ and $N \in \mathbb{N}$ are the initial time instant and the prediction horizon length over which the optimisation is carried out, respectively, $\mathscr{P}_k \in [\underline{\mathscr{P}}, \overline{\mathscr{P}}]$ (resp. $s_{f,k} \in [\underline{s}_f, \overline{s}_f]$) represents the admission probability (resp. the effective service time) attributed to the requests, and $\alpha \in [0,1]$ is a design parameter which allows to privilege either QoS or Energy saving objectives. Recall that $\mathscr{P}_k$ and $s_{f,k}$ are the control inputs to be computed by the LPV-MPC controller.

The cost function $J_N(\alpha,\bar{k})$ describes a convex combination of the performance objectives. More specifically, note that:

- $J_N(0,\bar{k}) = \sum_{k=\bar{k}}^{\bar{k}+N-1} \left| \frac{\overline{s}_f - s_{f,k}}{\overline{s}_f - \underline{s}_f} \right|$ represents the case where the objective is to maximise the effective service time associated to the requests, thus lowering the CPU frequency (Energy saving objective).

- $J_N(1,\bar{k}) = \sum_{k=\bar{k}}^{\bar{k}+N-1} \left| \frac{\overline{\mathscr{P}} - \mathscr{P}_k}{\overline{\mathscr{P}} - \underline{\mathscr{P}}} \right|$ represents the case where the objective is to maximise the admission probability, hence to satisfy the largest possible number of requests (QoS objective).

Note that the objective function is given in discrete time, where the sampling time $\Delta t \in \mathbb{R}^+$ is chosen according to the model description (here $\Delta t = 1$ min, see Section 3).

4.4 Dynamic Equality Constraints

The dynamic equality constraints of the considered problem are simply given by the LPV-IA model derived in Section 3. This results in a set of nonlinear dynamic constraints, denoted by $\Sigma(x,\lambda,\mathscr{P},s_f)$, namely

$$\begin{aligned} x_{k+1} &= Ax_k + \left(B_0 + B_1 s_{f,k} + B_2 s_{f,k}\lambda_k\right)\mathscr{P}_k \\ \xi_k &= Cx_k + \left(D_0 + D_1 s_{f,k} + D_2 s_{f,k}\lambda_k\right)\mathscr{P}_k \\ T_k &= \xi_k + s_{f,k}, \end{aligned} \quad (8)$$

where $A \in \mathbb{R}^{n\times n}$, $\{B_0,B_1,B_2\} \in \mathbb{R}^{n\times n_u}$, $C \in \mathbb{R}^{n_y\times n}$ and $\{D_0,D_1,D_2\} \in \mathbb{R}^{n_y\times n_u}$ are known matrices (here $n=2$, $n_u=1$ and $n_y=1$). Further, x_k represents the state of the system, and the response time T_k is the overall system output (see Equation (2)). Finally, λ_k and s_k, the latter needed to compute $s_{f,k}$, will be considered as known variables (up to time $\bar{k}+N-1$), even though, in practice, they are generally unknown (or only partially known thanks to some workload estimators).

4.5 Input and Performance Inequality Constraints

Now, let us define Λ as the set describing input and performance inequality constraints as follows

$$\Lambda : \begin{cases} 0 \le \xi_k \\ \underline{\mathscr{P}} \le \mathscr{P}_k \le \overline{\mathscr{P}} \\ \underline{s}_f \le s_{f,k} \le \overline{s}_f \\ -\Delta \le T_k - T_{ref} \le \Delta \end{cases} \quad (9)$$

where T_{ref} is the desired value of the response time and Δ defines the admissible tracking error $T_k - T_{ref}$. Both T_{ref} and Δ are design parameters. Note that, for simplicity, the desired value of the response time T_{ref} has been defined as constant. In the same way, the proposed approach can handle a time-varying set-point trajectory, i.e., $T_{ref,k}$, which may be useful to handle QoS requirements which are functions, for example, of the time of day.

4.6 LPV-MPC Optimisation Problem

The considered constrained finite-time optimal control problem to be solved at each sampling time $\bar{k}$ is defined as

$$J_N^*(\alpha,\bar{k}) = \min J_N(\alpha,\bar{k})$$
$$\text{subject to} \begin{cases} \begin{bmatrix} x_{k+1} \\ \xi_k \\ T_k \end{bmatrix} = (8) \\ \Lambda \quad = (9) \end{cases} \quad k \in [\bar{k}, \bar{k}+N-1].$$

This problem is iteratively solved using the YALMIP parser and a standard nonlinear constrained solver via interior point methods [40, 41]. The Reader should keep in mind that the proposed optimal performance computation is *ideal*, as perfect modeling, full state and disturbances knowledge are assumed. Still, this approach provides an upper bound on the achievable server performance and can help both dynamic performance evaluation and controller design. As mentioned before, in fact, the ideal controller can be easily adapted to be employed on the real system via state estimation and assuming no knowledge of the disturbance (or complementing the system with workload predictors).

In the following, the LPV-MPC controller is used to evaluate the optimal performance of the considered application, based on the dynamical model identified in Section 3, both via time domain simulations and by analysing the performance index (7), which is representative of the QoS/Energy saving trade-off.

Digital Object Identifier: 10.4108/ICST.VALUETOOLS2009.7941
http://dx.doi.org/10.4108/ICST.VALUETOOLS2009.7941

5. SIMULATION RESULTS

In this Section, simulations are carried out to analyse the influence of the parameter α in the cost function (7), i.e., the trade-off between QoS and Energy saving. To perform the analysis we assume that the system is subject to the requests rate shown in Figure 2.

Additionally, in order to quantitatively evaluate both QoS and Energy saving objectives, the following performance measures will be employed

$$\begin{aligned} P_{QoS} &= \sum_{k=1}^{N_f} \frac{||\mathscr{P}_k - \underline{\mathscr{P}}||_2}{||\overline{\mathscr{P}} - \underline{\mathscr{P}}||_2} \\ P_{ES} &= \sum_{k=1}^{N_f} \frac{||s_{f,k} - \underline{s}_f||_2}{||\overline{s}_f - \underline{s}_f||_2} \end{aligned} \tag{10}$$

where N_f is the final simulation sample. As a consequence:

- $P_{QoS} = 1$ (resp. 0) indicates that the maximum (resp. the minimum) number of requests have been served with the desired response time T_{ref}.
- $P_{ES} = 1$ (resp. 0) indicates that the power consumption of the server have been minimised (resp. maximised), i.e., the server CPU frequency has been lowered (resp. increased).

To perform the simulations, the following server configuration is considered:

- The service time limits are chosen as $[\underline{s}_f, \overline{s}_f] = [0.5\,s_k, s_k]$. This choice illustrates the fact that when $s_{f,k} = s_k$ the server runs at a low frequency, while, when $s_{f,k} = 0.5\,s_k$ the server frequency must double, thus consuming more energy. Note that these bounds can be viewed as a *system parameter*, as they depends on the DVS settings of the server in use.
- The admission probability limits are chosen as $[\underline{\mathscr{P}}, \overline{\mathscr{P}}] = [0.5, 1]$. This configuration implies that the server system is at least constrained to accept 50% of the incoming requests. Note that this parameter is a *design parameter* since it depends on the provider/customer QoS negotiation.
- The reference response time has been set to $T_{ref} = 1$ s and the maximum tracking error to $\Delta = \frac{5}{100} T_{ref}$.
- The prediction horizon length has been set to $N = 20$. This variable modifies the optimization problem by enlarging the time interval over which the controller has knowledge of the future disturbance evolution.

We now move to the analysis of the QoS/Energy trade-off. At first, the performance measures (10) are evaluated for different values of α. Figure 6 shows the values of P_{Qos} and P_{ES} in (10) for different values of α.

By inspecting Figure 6, the trade-off between QoS and energy saving is apparent. When α is low (see also Equation (7)) energy saving is a priority. Conversely, increasing α leads to higher energy consumption but allows a larger number of requests to be admitted in the system, thus maximising QoS. According to these results, the following remarks are due.

- When $\alpha = 0$, P_{ES} is maximised, while the P_{QoS} is almost null, which means that the admission probability is almost always at its lower bound of $\underline{\mathscr{P}} = 0.5$.
- When $\alpha = 1$, the P_{ES} is minimised, while P_{QoS} is maximised, which means that the admission probability is almost always at its upper bound $\overline{\mathscr{P}} = 1$ and the effective service time is low.

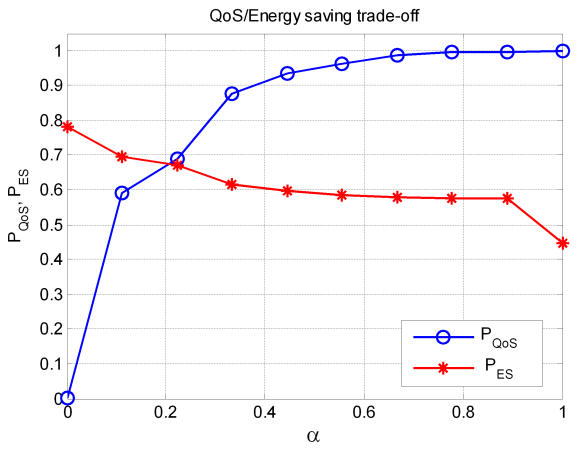

Figure 6: Plot of the performance measures P_{Qos} and P_{ES} as functions of α for $N = 20$.

- The maximal and minimal values of the P_{ES} curve are not reached. This is due to the fact that, according to the constraints described in the previous section, it is not possible for the system to guarantee an admission probability of $\mathscr{P}_k = \underline{\mathscr{P}} = 0.5$ while maximising the effective service time $s_{f,k}$. This should be possible by e.g., reducing the minimal admission probability or by increasing the desired response time value T_{ref}. This highlights the flexibility of the proposed approach, which allows to analyse the system performance for different designer choices, thereby providing a valuable means to evaluate different system settings.
- A reasonable trade-off between QoS and Energy saving is achieved, in our setting, for $\alpha = 0.25$.

Finally, to illustrate the proposed LPV-MPC controller performance analysis on some representative simulation cases, the following examples are considered:

- Simulation 1: $N = 20$ and $\alpha = 0$; Energy saving objective.
- Simulation 2: $N = 20$ and $\alpha = 0.25$; Trade-off between Energy saving and QoS.

The optimal control-based server performance is compared with an open loop configuration (e.g., with no control) where both the effective service time and the admission probability are fixed as follows

- The service time is set to $s_{f,k} = 0.75\,\overline{s}_k$, which means that the server works at a nominal CPU frequency and all requests have a constant service time. This simplifying assumption allows to better highlight the advantages of the closed-loop solution with respect to the one in open loop, by evaluating the latter in a quite favorable situation.
- The admission probability is set to $\mathscr{P}_k = \overline{\mathscr{P}} = 1$, which is equivalent to accepting 100% of the incoming requests.

The obtained results are shown in Figures 7 and 8. Specifically, the top plots of Figures 7 and 8 both confirm that the constraint on the response time T_k is always fulfilled when the system is controlled. Moreover, these results also show that the uncontrolled

Digital Object Identifier: 10.4108/ICST.VALUETOOLS2009.7941
http://dx.doi.org/10.4108/ICST.VALUETOOLS2009.7941

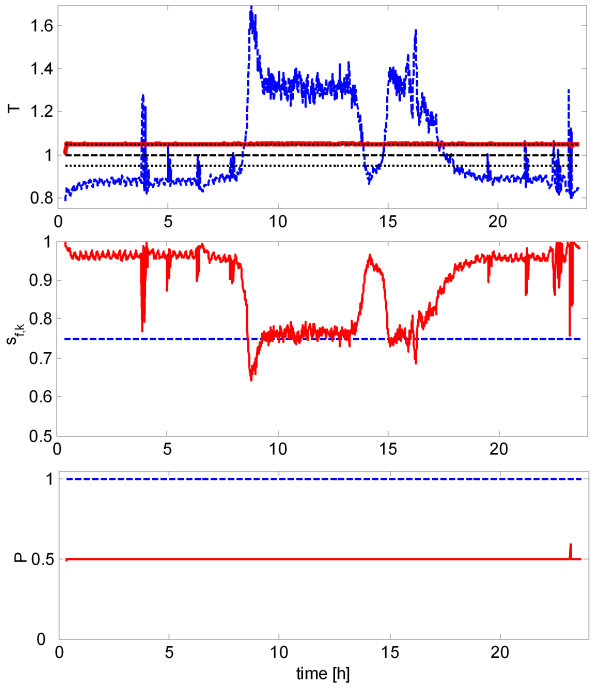

Figure 7: Simulation study 1. Top plot: response time T_k. Middle plot: effective service time, $s_{f,k}$. Bottom plot: admission probability $\mathscr{P}_k$. Uncontrolled server (dashed line), Nonlinear optimal LPV-MPC (solid line).

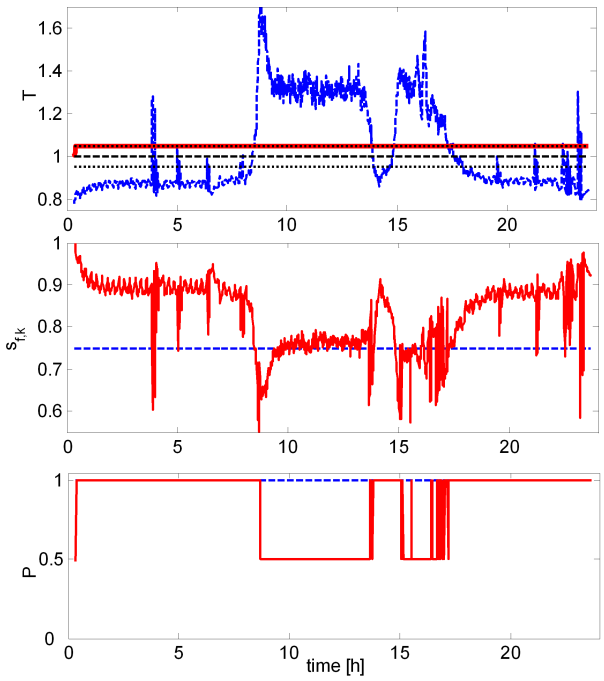

Figure 8: Simulation study 2. Top plot: response time T_k. Middle plot: $s_{f,k}$, effective service time. Bottom plot: $\mathscr{P}_k$, admission probability. Uncontrolled server (dashed line), Nonlinear optimal MPC (solid line).

server provides very poor performance when the number of requests increases, leading to poor QoS. This confirms the need of using a controller to enhance the system performance.

Further, the middle and bottom plot of Figures 7 and 8 prove that input and performance constraints are satisfied. Additionally, they show how the admission probability and effective service time are adjusted to guarantee a good level of both QoS and energy saving. More specifically:

- Figure 7 (Simulation 1, Energy saving objective) shows that, over the whole experiment, the admission probability $\mathscr{P}_k$ (bottom plot) is kept constant at its lower bound, whilst $s_{f,k}$ (middle plot) is modulated to maximise energy saving (consistently with the settings of the performed simulation test). As a matter of fact, over the time interval in which λ_k is low (see top plot), the effective service time is increased, meaning that the CPU frequency is lowered, thus yielding energy saving.

- Figure 8 (Simulation 2, Qos/Energy saving trade-off objective) shows that, when the number of requests is large, i.e., between 9 and 16 h, the admission probability $\mathscr{P}_k$ (bottom plot) is reduced to allow obtaining an effective service time $s_{f,k}$ as large as possible (middle plot), in order to save energy. Conversely, over the time interval in which λ_k is low the effective service time is increased, meaning that the CPU frequency is lowered, thus resulting in reduced energy consumption and the admission probability $\mathscr{P}_k$ is at its upper bound to ensure good QoS. Indeed, since in this simulation the aim was to ensure a trade-off between QoS and energy saving, this behaviour is consistent with the objectives.

6. CONCLUDING REMARKS

This paper presented a control theoretic framework for the dynamic analysis of QoS/Energy trade-offs in Web service systems. Specifically, the contribution is twofold: first, an effective identification approach for modeling admission control dynamics has been presented. Secondly, a performance analysis method to evaluate the achievable server performance has been proposed using a numerical optimization approach involving the estimated LPV dynamical model and an iterative optimization procedure. This problem formulation leads to an *optimal closed-loop* performance analysis illustrating both the QoS/Energy saving trade-off and the interest in controlling both the service time and the admission rate to improve performance while guaranteeing a given desired service response time. Further, the analysis of such an optimal control solution can be very useful to design an actual LPV-MPC controller, as it allows to analyse its performance against such a benchmark so as to quantitatively evaluate its degree of sub-optimality at design time.

Future work will address the design of real-time LPV-MPC controllers by complementing the control system with state estimation and workload prediction capabilities.

7. REFERENCES

[1] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *IEEE Computer*, vol. 36, no. 1, pp. 41–50, 2003.

Digital Object Identifier: 10.4108/ICST.VALUETOOLS2009.7941
http://dx.doi.org/10.4108/ICST.VALUETOOLS2009.7941

[2] V. Metha, "A Holistic Solution to the IT Energy Crisis," 2007. [Online]. Available: http://greenercomputing.com/

[3] J. S. Chase and D. C. Anderson, "Managing Energy and Server Resources in Hosting Centers," in *ACM Symposium on Operating Systems principles*, 2001.

[4] J. S. T. Abdelzaher and, C. Lu, R. Zhang, and Y. Lu, "Feedback Performance Control in Software Services," *IEEE Control Systems Magazine*, vol. 23, no. 3, pp. 21–32, 2003.

[5] D. Kusic, J. O. Kephart, N. Kandasamy, and G. Jiang, "Power and Performance Management of Virtualized Computing Environments Via Lookahead Control," in *ICAC 2008 Proc.*, 2008.

[6] W. Qin and Q. Wang, "Modeling and control design for performance management of web servers via an LPV approach," *IEEE Transactions on Control Systems Technology*, vol. 15, no. 2, pp. 259–275, 2007.

[7] D. Kusic and N. Kandasamy, "Risk-Aware Limited Lookahead Control for Dynamic Resource Provisioning in Enterprise Computing Systems," in *ICSOC 2004 Proc.*, 2004.

[8] J. Kephart, H. Chan, R. Das, D. Levine, G. Tesauro, F. Rawson, and C. Lefurgy, "Coordinating Multiple Autonomic Managers to Achieve Specified Power-performance Tradeoffs," in *ICAC Proceedings*, June 2007.

[9] J. Carlstrom and R. Rom, "Application-aware Admission Control and Scheduling in Web Servers," in *Infocom Proc.*, 2002.

[10] M. Welsh and D. Culler, "Adaptive overload control for busy internet servers," in *USITS Proceedings*, 2003.

[11] B. Urgaonkar and P. Shenoy, "Sharc: Managing CPU and Network Bandwidth in Shared Clusters," *IEEE Transactions on Parallel and Distributed Systems*, vol. 15, no. 1, pp. 2–17, 2004.

[12] G. Pacifici, M. Spreitzer, A. N. Tantawi, and A. Youssef, "Performance Management for Cluster-Based Web Services," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 12, pp. 2333–2343, December 2005.

[13] B. Urgaonkar, G. Pacifici, P. J. Shenoy, M. Spreitzer, and A. N. Tantawi, "Analytic modeling of multitier Internet applications," *ACM Transaction on Web*, vol. 1, no. 1, January 2007.

[14] L. Zhang and D. Ardagna, "SLA Based Profit Optimization in Autonomic Computing Systems," in *ICSOC 2004 Proceedings*, New York, 2004, pp. 173–182.

[15] B. Abrahao, V. Almeida, J. Almeida, A. Zhang, D. Beyer, and F. Safai, "Self-Adaptive SLA-Driven Capacity Management for Internet Services," in *Proc. NOMS06*, 2006.

[16] P. Apkarian, P. Gahinet, and G. Becker, "Self-scheduled H_∞ control of linear parameter-varying systems," *Automatica*, vol. 31, no. 9, pp. 1251–1261, 1995.

[17] J. S. Shamma and M. Athans, "Analysis of Gain-Scheduled Control for Nonlinear Plants," *IEEE Transactions on Automatic Control*, vol. 35, no. 8, pp. 898–907, 1990.

[18] M. Tanelli, D. Ardagna, M. Lovera, and L. Zhang, "Model Identification for Energy-Aware Management of Web Service Systems," in *ICSOC08 Proc.*, 2008.

[19] M. Tanelli, D. Ardagna, and M. Lovera, "LPV Model Identification in Virtualized Service Center Enviroments," in *15th IFAC Symposium on System Identification, Saint Malo, France*, 2009.

[20] E. Camacho and C. Bordons, *Model Predictive Control*. New York: Springer Verlag, 1999.

[21] J. Maciejowski, *Predictive Control with Constraints*. Prentice-Hall, 2001.

[22] L. Kleinrock, *Queueing Systems*. John Wiley and Sons, 1975.

[23] W. Qin and Q. Wang, "An LPV approximation for admission control of an internet web server: identification and control," *Control Engineering Practice*, vol. 15, no. 12, pp. 1457–1467, 2007.

[24] R. Toth, F. Felici, P. Heuberger, and P. V. den Hof, "Discrete time LPV I/O and state space representations, differences of behavior and pitfalls of interpolation," in *Proceedings of the 2007 European Control Conference, Kos, Greece*, 2007.

[25] M. Verhaegen, "Identification of the deterministic part of MIMO state space models given in innovations form from input output data," *Automatica*, vol. 30, no. 1, pp. 61–74, 1994.

[26] L. Lee and K. Poolla, "Identification of linear parameter-varying systems using nonlinear programming," *ASME Journal of Dynamic Systems, Measurement and Control*, vol. 121, no. 1, pp. 71–78, 1999.

[27] V. Verdult, "Nonlinear system identification: A state-space approach," Ph.D. dissertation, University of Twente, Faculty of Applied Physics, Enschede, The Netherlands, 2002.

[28] Apache, "Apache JMeter." [Online]. Available: http://jakarta.apache.org/jmeter/index.html

[29] A. Williams, M. Arlitt, C. Williamson, and K. Barker, *Web Workload Characterization: Ten Years Later*, ser. Web Information Systems Engineering and Internet Technologies Book Series. Springer US, 2005, pp. 3–21.

[30] N. Mi, G. Casale, L. Cherkasova, and E. Smirni, "Injecting realistic burstiness to a traditional client-server benchmark," in *ICAC '09: Proceedings of the 6th international conference on Autonomic computing*, 2009, pp. 149–158.

[31] Apache, "Apache Tomcat." [Online]. Available: http://tomcat.apache.org/

[32] The Open Group, "Application Resource Measurement - ARM." [Online]. Available: http://www.opengroup.org/tech/management/arm/

[33] L. Magni and R. Scattolini, "Robustness and robust design of MPC for nonlinear discrete-time systems," *Lecture Notes in Control and Information Sciences*, vol. 358, pp. 239–254, 2007.

[34] G. De Nicolao, L. Magni, and R. Scattolini, "Stabilizing receding-horizon control of nonlinear time-varying systems," *IEEE Trans. on Automatic Control*, vol. AC-43, pp. 1030–1036, 1998.

[35] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, pp. 789–814, 2000.

[36] T. Besselmann, J. Lofberg, and M. Morari, "Explicit Model Predictive Control for Linear Parameter-Varying Systems," in *47th IEEE Conference on Decision and Control, 2008.*, 2008, pp. 3848–3853.

[37] Y. Lu and Y. Arkun, "Quasi-Min-Max MPC algorithms for LPV systems," *Automatica*, vol. 36, pp. 527–540, 2000.

[38] L. Chisci, P. Falugi, and G. Zappa, "Gain-scheduling MPC of nonlinear systems," *International Journal of Robust and Nonlinear Control*, vol. 13, no. 3-4, pp. 295–308, 2003.

[39] M. Andreolini and S. Casolari, "Load prediction models in

Digital Object Identifier: 10.4108/ICST.VALUETOOLS2009.7941
http://dx.doi.org/10.4108/ICST.VALUETOOLS2009.7941

web-based systems," in *Proceedings of the 1st International Conference on Performance Evaluation Methodologies and Tools, Pisa, Italy*, 2006.

[40] J. Lofberg, "YALMIP : A toolbox for modeling and optimization in MATLAB," in *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004. [Online]. Available: http://control.ee.ethz.ch/ joloef/yalmip.php

[41] T. Coleman and Y. Zhang, *Optimization Toolbox (for use with MATLAB)*, v2.11 ed., June 2001.

Digital Object Identifier: 10.4108/ICST.VALUETOOLS2009.7941
http://dx.doi.org/10.4108/ICST.VALUETOOLS2009.7941