16th Mediterranean Conference on Control and Automation
Congress Centre, Ajaccio, France
June 25-27, 2008

Robust Almost Sure Stability for Uncertain Stochastically Scheduled Anytime Controllers

Luca Greco, Daniele Fontanelli and Antonio Bicchi

Abstract— **In this paper we consider closed loop stability of a number of different software tasks implementing a hierarchy of real-time controllers for a given plant, i.e. "anytime controllers". The execution of the control tasks are driven by the available computational time of an embedded platform under stringent real-time constraints. Hence, preemptive scheduling schemes are considered, under which the maximum execution time allowed for control software tasks is not a–priori known. A stochastic description of the scheduler accounting for the presence of some uncertainties in the model, is provided. An anytime control hierarchy of controllers for the same plant, in which higher controllers in the hierarchy provide better closed–loop performance but require larger worst-case execution times, is assumed to be given. Since the ensuing switching system is prone to instability, the presented method allows to robustly condition the partially known stochastic scheduler so as to obtain a better exploitation of computing capabilities, while guaranteeing almost sure stability of the resulting switching system.**

I. INTRODUCTION

In the implementation of controllers on an embedded programmable processor, it is often the case that real-time tasks have to concurrently share computational resources with several other tasks, thus reducing the overall HW cost and development time. Among such tasks, those implementing control algorithms are usually highly time critical, and have traditionally imposed very conservative scheduling approaches, whereby execution time is allotted statically, which makes the overall architecture extremely rigid, hardly reconfigurable for additions or changes of components, and often underperforming. In modern applications, the available computational time is scheduled dynamically, using preemptive scheme to adapt to varying load conditions and Quality of Service requirements. Representative examples of such scheduling algorithms are Rate Monotonic (RM) and Earliest Deadline First (EDF) [1], [2], [3], whose schedulability guarantee is provided based on estimates of the Worst-Case Execution Time (WCET) of tasks. Nevertheless, as stated for example in [1] where it is shown that RM scheduling can meet all deadlines if the CPU utilization is not larger than 69.3%, conservative assumptions on the computational power budget, control tasks complexity and costs are usually needed to make a given set of tasks schedulable and to limit the number of deadline misses.

Substantial performance improvement would be gained if less conservative assumptions could be made on the CPU utilization. In particular, it is often the case that, for most of the CPU cycles, a time τ could be made available for a control task which is substantially longer than τ_{min}, although only the latter can be guaranteed in the worst case. In [4] the authors proposed a synthesis methodology to design controllers, referred to as *anytime controllers*, capable to guarantee a useful result whenever the algorithm is run for at least τ_{min} and to provide better results if longer times are allowed. The key idea is borrowed from so-called anytime algorithms, that have been proposed in real-time computation ([5]). The characteristic of anytime algorithms (or of *imprecise computation*, as they are sometimes referred to) is to always return an answer on demand; however, the longer they are allowed to compute, the better (e.g. more precise) an answer they will return. In digital filter design [6], this philosophy has been pursued by decomposing the full-order filter in a cascade of lower order filters whose execution is prioritized.

In the control theory domain, a periodic task is split in a mandatory part, schedulable in τ_{min}, and in one or more optional parts ([4]). Therefore, a classical monolithic control task is replaced by a hierarchy of control tasks of increasing complexity, each providing a correspondingly increasing "quality of control". The ensuing closed loop system is then switching, where the switches are driven by the preemptive events of the scheduler. The substantial literature on *switching system* stability (see e.g. [7], [8], [9] and references therein) provides much inspiration and ideas for the problem at hand, but few results can be used directly. For instance, [10] provides a method to design controllers ensuring stability under arbitrary switching sequence, but the computational complexity of each controller and the necessity of reset maps at any switching instant make its method not suited for the anytime approach. By the same practicality argument, algorithms for switched system stabilization (such as e.g. [11], [12], [13]) requiring the computation of complex functions of the state to ascertain which subsystem can be activated next time, are not applicable to anytime control.

On the other hand, in [14] a framework for the stability analysis of anytime control algorithms, based on a stochastic model for the scheduler, is presented and a switching policy capable of conditioning the stochastic properties of the scheduler was designed, such that overall stability (in the probabilistic sense of "almost sure" stability [15], [16]) of

D. Fontanelli and A. Bicchi are with the Interdept. Research Center "E. Piaggio", University of Pisa, Italy. This work was supported by the EC under contract IST 511368 (NoE) "HYCON - Hybrid Control: Taming Heterogeneity and Complexity of Networked Embedded Systems" and contract IST 045359 "PHRIENDS" - Physical Human-Robot Interaction: Dependability and Safety"

L. Greco is with the University of Salerno, Italy. This work has been (partially) supported by Cassa di Risparmio di Pisa, Lucca e Livorno within the project "Progetto di Ricerca 2006" of the University of Pisa.

the resulting Markov Jump Linear System (MJLS) can be guaranteed. This paper extends the results of [14] allowing a stabilizing stochastic switching policy to be provided even when scheduler's features are fixed but not perfectly known. In this case, the scheduler can be considered affected by some uncertainties. If they affect directly the steady-state probability distribution of the scheduler and are bounded by a polytopic set, the linear programming problem, looking for a stabilizing policy, can be easily extended. Instead, when it is the transition probability matrix to be uncertain, the robust problem formulation is more complex. Indeed, a polytopic set including the set of steady-state probability distributions is not directly available and must be computed. We provide an iterative algorithm capable of yielding such a polytopic approximation, which has some similarities with the algorithm used in the construction of the polytopic invariant set for a polytope of Schur matrices ([17], [18]). The algorithm starts with the entire set of the probability distributions and refines its approximation at each step, thus providing a valid set for which solving the robust stability problem. The effectiveness of the proposed approach is shown with a regulation example on a Furuta pendulum ([19]).

II. Scheduling Problem Formulation and Controller Design

Let $\Sigma \triangleq (A, B, C)$ be the given strictly proper linear, discrete time, invariant plant to be controlled, and let $\Gamma_i \triangleq (F_i, G_i, H_i, L_i)$, $i \in I \triangleq \{1, 2, \ldots, n\}$ be a family of feedback controllers for Σ. Assume that all controllers Γ_i stabilize Σ and are ordered by increasing computational time complexity, i.e. $WCET_i > WCET_j$ if $i > j$. Let the closed-loop systems thus obtained be $\Sigma_i \triangleq (\widehat{A}_i, \widehat{B}_i, \widehat{C}_i)$, where

$$\widehat{A}_i = \begin{bmatrix} A + BL_iC & BH_i \\ G_iC & F_i \end{bmatrix};$$
$$\widehat{B} = \begin{bmatrix} B_i \\ 0 \end{bmatrix}; \widehat{C}_i = \begin{bmatrix} C_i & 0 \end{bmatrix}.$$

Problems related to jitter and delay are not considered in this work since they can be tackled in the design of the single controllers ([20]). Therefore, we assume that measurements are acquired and control inputs are released at every sampling instant tT_g, $t \in \mathbb{N}$, where T_g is a fixed sampling time. Let $\gamma_t \in [\tau_{min}, \tau_{max}]$, $\tau_{max} < T_g$, denote the time allotted to the control task during the t-th sampling interval. By hypothesis, $WCET_1 \leq \tau_{min}$ and $WCET_n \leq \tau_{max}$.

Define an event set $L_\tau \triangleq \{\tau_1, \ldots, \tau_n\}$, and a map

$$\begin{array}{rcccl} \mathcal{T} & : & [\tau_{min}, \tau_{max}] & \rightarrow & L_\tau \\ & & \gamma_t & \mapsto & \tau(t) \end{array}$$

where

$$\tau(t) = \begin{cases} \tau_1, & \text{if} \quad \gamma_t \in [\tau_{min}, WCET_2) \\ \tau_2, & \text{if} \quad \gamma_t \in [WCET_2, WCET_3) \\ \vdots & \text{if} \quad \vdots \\ \tau_n, & \text{if} \quad \gamma_t \in [WCET_n, \tau_{max}] \end{cases}$$

Assume a stochastic description of the scheduling process to be given by

$$\Pr\{\tau(t) = \tau_i\} = \overline{\pi}_{\tau_i}, \; 0 < \overline{\pi}_{\tau_i} < 1, \; \sum_{i \in I} \overline{\pi}_{\tau_i} = 1,$$

where $\overline{\pi}_{\tau_i}$ denotes the probability associated to the event that the time slot γ_t is such that all controllers $\Gamma_j, j \leq i$, but no controller $\Gamma_k, k > i$, can be executed. The distribution $\overline{\pi}_\tau = [\overline{\pi}_{\tau_1}, \overline{\pi}_{\tau_2}, \cdots, \overline{\pi}_{\tau_n}]^T$ can be regarded simply as an i.i.d. process, or, in a slightly more complex but general way, as the invariant probability distribution of a finite state discrete-time homogeneous irreducible aperiodic Markov chain given by

$$\pi(t+1) = P^T \pi(t), \; \pi(0) = \pi_0.$$

where $P = (p_{ij})_{n \times n}$ is the transition probability matrix and p_{ij} is the transition probability from state i to state j of the Markov chain (e.g. from controller Γ_i to Γ_j). Under these hypotheses, the switching process generates a discrete-time Markov Jump Linear System (MJLS)

$$x_{t+1} = \widehat{A}_{\tau_t} x_t \tag{1}$$

Definition 1: [16] The MJLS (1) is said almost surely stable (AS-stable) if there exists $\mu > 0$ such that, for any $x_0 \in \mathbb{R}^N$ and any initial distribution π_0, the following condition holds

$$\Pr\left\{\limsup_{t \to \infty} \frac{1}{t} \ln \|x_t\| \leq -\mu\right\} = 1.$$

Let $\|\cdot\|$ be a matrix norm induced by some vector norm. The following sufficient condition for AS-stability was proved in [15]:

Theorem 1 (1–step average contractivity): [15] If

$$\xi_1 = \prod_{i \in I} \|\widehat{A}_{\tau_i}\|^{\overline{\pi}_{\tau_i}} < 1 \tag{2}$$

then the MJLS (1) is AS-stable.

A. Design of a Control Algorithm Hierarchy

In this section we briefly resume a bottom-up design technique, based on classical cascade design, to determine an ordered set of control algorithms providing increasing closed-loop performance (see [4] for a more comprehensive discussion on controller design in the anytime framework). Consider the two design stages illustrated in fig. 1, in which controllers are designed to ensure increasing performance by any classical synthesis technique. The scheme in fig. 1 cannot be implemented as a composable anytime control, because after computation of the a) scheme, the input to the $F_1(z)$ block needs to be recomputed completely if the b) scheme is to be applied. However, by simple block manipulations, the scheme in fig. 2 can be obtained, where we set

$$\hat{C}_2(z) = F_1(z) C_2(z).$$

The scheme in fig. 2 is suitable for anytime implementation. Indeed the series of $F_1(z)$ and $F_2(z)$ is in open-loop (hence equivalent to an anytime filter), while the parallel connection in the feedback loop is simply obtained by summing the new result by $\hat{C}_2(z)$ to the previous one by $C_1(z)$.

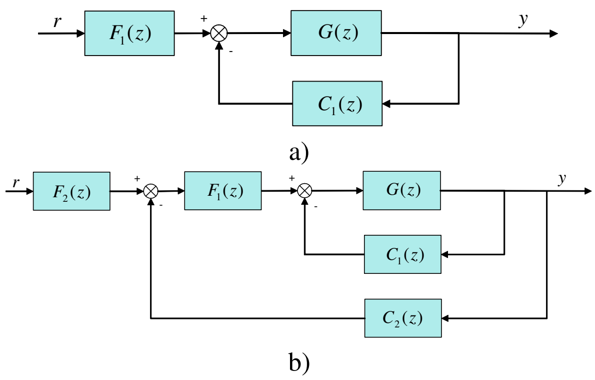


Fig. 1. Two stages of a classical cascade design procedure

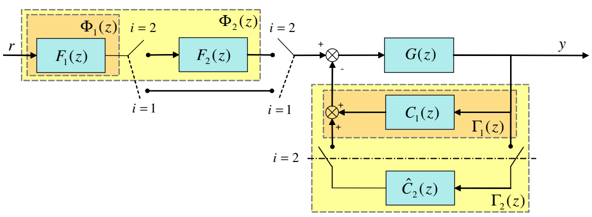


Fig. 2. A switched control scheme suitable for anytime control implementation. The scheme is equivalent to fig. 1-a when the switches are in the $i = 1$ position, and to fig. 1-b for $i = 2$.

III. Stochastic Schedule Conditioning

We define a *switching policy* to be a map $s : \mathbb{N} \rightarrow I$, $t \mapsto s(t)$, which determines an upper bound to the index i of the controller to be executed at time t, i.e. $i \leq s(t)$. In other terms, at time tT_g, the system starts computing the controller algorithm until it can provide the output of $\Gamma_{s(t)}$, unless a preemption event occurs forcing it to provide only $\Gamma_{\tau(t)}$, i.e. the highest controller computed before preemption. Application of a switching policy s to a set of feedback systems Σ_i, $i \in I$ under a scheduler τ generates a switching linear system (Σ_i, τ, s) which, under suitable hypotheses, is also a MJLS. The stochastic characterization of the chain τ is assumed to be a-priori known. Furthermore, in a real application (e.g. automotive domain) different working conditions lead to different stochastic descriptions, thus different Markov chains for the scheduler can be considered.

As an example, the most conservative policy is to set $s(t) \equiv 1$, i.e. forcing always the execution of the simplest controller Γ_1, regardless of the probable availability of more computational time. By assumption, this (non-switching) policy guarantees stability of the resulting closed loop system.

On the opposite, a "greedy" strategy would set $s(t) \equiv n$, which leads to providing $\Gamma_{\tau(t)}$ for all t. Although this policy attempts at maximizing the utilization of the most performing controller, it is well known that switching arbitrarily among asymptotically stable systems Σ_i may easily result in an unstable behavior [21].

A sufficient condition for the greedy switching policy to provide an AS-stable system is provided by Theorem 1. This condition however is rarely satisfied. Indeed, the fact that each matrix $\widehat{A}_{\tau_i}$ is Schur guarantees the existence of a specific norm $\|\cdot\|_{w_i}$ such that $\|\widehat{A}_{\tau_i}\|_{w_i} < 1$, but no single norm $\|\cdot\|_w$ exist in general such that $\|\widehat{A}_{\tau_i}\|_w < 1 \ \forall \tau_i$[1]. The AS stability condition of theorem 1 would require that, for a chosen norm, for all controllers with $\|\widehat{A}_{\tau_i}\|_w > 1$ $\overline{\pi}_{\tau_i}$ is sufficiently small, i.e. they are scheduled by the OS sufficiently rarely.

A switching policy that suitably conditions the scheduler to provide AS-stability was studied in [14], which is illustrated below. Introduce a homogeneous irreducible aperiodic Markov chain σ with the same number n of states as the scheduler chain τ. The states are labelled as σ_i, with the meaning that if the associated process form $\sigma(t)$ is equal to σ_i, then $s(t) = i$, i.e. in the next sampling interval tT_g at most the i–th controller is computed (this will actually happen if no preemption occurs). We will refer to σ as the conditioning Markov chain. The synthesis of such a conditioning Markov chain can be formulated as the following Linear Programming problem:

$$\begin{array}{ll} \multicolumn{2}{l}{\text{Find a vector } \overline{\pi}_\sigma = \begin{bmatrix} \overline{\pi}_{\sigma_1} & \cdots & \overline{\pi}_{\sigma_n} \end{bmatrix}^T \text{ such that}} \\ 1) & (M_c \overline{\pi}_\tau)^T \overline{\pi}_\sigma < 0 \\ 2) & 0 < \overline{\pi}_{\sigma_i} < 1 \\ 3) & \displaystyle\sum_{i=1}^{n} \overline{\pi}_{\sigma_i} = 1, \end{array} \tag{3}$$

where

$$M_c = \begin{bmatrix} \ln\left(\left\|\widehat{A}_{\min(\tau_1,\sigma_1)}\right\|\right) & \cdots & \ln\left(\left\|\widehat{A}_{\min(\tau_n,\sigma_1)}\right\|\right) \\ \vdots & \cdots & \vdots \\ \ln\left(\left\|\widehat{A}_{\min(\tau_1,\sigma_n)}\right\|\right) & \cdots & \ln\left(\left\|\widehat{A}_{\min(\tau_n,\sigma_n)}\right\|\right) \end{bmatrix}.$$

Should this problem not have a feasible solution, multi–step switching policies can be considered, whereby the conditioning Markov chain suggests the sequence of controllers to be executed in the next m steps (see [14] for details). This way, some control patterns, i.e. substrings of symbols in I, are preferentially used with respect to others.

IV. Robustness (1-step)

To start with the robustness problem, let us assume uncertainties affect the steady-state vector $\overline{\pi}_\tau$. A description for uncertain but fixed parameters is often provided in terms of a polytopic set with a finite number of vertices. In this case the unknown $\overline{\pi}_\tau$ can be considered belonging to the vector polytope $\boldsymbol{\pi} = \text{conv}\left\{\pi_\tau^1, \ldots, \pi_\tau^r\right\}$. This description provides a simple way to manage the robustness problem. Indeed, the problem (3) is linear, hence convex, w.r.t. $\overline{\pi}_{\tau_h}$, therefore, a $\overline{\pi}_\sigma$ solution of the stability problem for every $\overline{\pi}_\tau \in \boldsymbol{\pi}$ can be found simply by replacing the first inequality in (3) with r inequalities, one for each vertex π_τ^i.

If the uncertainties are considered affecting directly the transition probability matrix P of the scheduler, the robustness problem needs a more complex formulation. We can consider the unknown but fixed matrix P to belong to

[1]When this happens, such norm is a common Lyapunov function and the system remains stable for all switching sequences. It is well known that this is rarely the case.

a stochastic matrix polytope $\mathcal{P} = \text{conv}\{P_1, \dots, P_m\}$. In order to solve the robustness linear programming problem, however, we must define the set of uncertain steady-state probability distributions $\overline{\pi}_\tau$ generated by $\mathcal{P}$. Unfortunately, the equation $\overline{\pi}_\tau^T(P - I) = 0$ relating P and its steady-state probability distribution $\overline{\pi}_\tau$, is not convex, hence even the set

$$\mathcal{L} = \begin{cases} \overline{\pi}_\tau^T(P - I) = 0, & P \in \mathcal{P} \\ 0 < \overline{\pi}_{\tau_i} < 1 \\ \sum_i \overline{\pi}_{\tau_i} = 1 \end{cases}$$

is, in general, not convex too. This fact implies that we cannot describe the uncertainty set $\boldsymbol{\pi}$ by means of steady-state vectors $\overline{\pi}_\tau^i$ of the vertex matrices P_i, since, in general, $\text{conv}\{\overline{\pi}_\tau^1, \dots, \overline{\pi}_\tau^m\} \not\supseteq \mathcal{L}$.

A possible way of addressing this problem is to build a finite vector polytope $\boldsymbol{\pi}$ such that $\boldsymbol{\pi} \supseteq \mathcal{L}$.

Before proceeding in the construction of $\boldsymbol{\pi}$, it is worth noting that such a polytope exists. In fact the simplex

$$\mathcal{S} = \begin{cases} 0 \le \overline{\pi}_{\tau_i} \le 1 \\ \sum_i \overline{\pi}_{\tau_i} = 1 \end{cases}$$

is, obviously, a finite polytope including $\mathcal{L}$, as it defines the overall probability distribution set.

Consider now any set $W \subseteq \mathcal{S}$ and its mapping along $\mathcal{P}$

$$\mathcal{P}W = \text{conv}\{P_1 W, \dots, P_m W\}.$$

As a consequence of the linearity of the mapping, if W is a polytope given by $W = \text{conv}\{w^1, \dots, w^r\}$, then $\mathcal{P}W$ is a polytope given by

$$\mathcal{P}W = \underset{\substack{i=1,\dots,m \\ j=1,\dots,r}}{\text{conv}} \{P_i w^j\}.$$

The set $\mathcal{L}$ is made up of all the fixed points for the mapping $\mathcal{P}$, that is $\forall \overline{\pi}_\tau \in \mathcal{L}$, $\exists P \in \mathcal{P}$ such that $\overline{\pi}_\tau^T P = \overline{\pi}_\tau^T$. Hence, by the definition of $\mathcal{L}$, we have that $\mathcal{P}\mathcal{L} \supseteq \mathcal{L}$. For the same reason, if W is a polytope such that $W \supseteq \mathcal{L}$, then $\mathcal{P}W \supseteq \mathcal{L}$. It is worth noting that in general $\mathcal{P}W \not\supseteq W$, but being $W \supseteq \mathcal{L}$ and $\mathcal{P}W \supseteq \mathcal{L}$, it is apparent that $\mathcal{P}W \cap W \supseteq \mathcal{L}$. This fact suggests the following iterative algorithm to build a tight polytope $\boldsymbol{\pi} \supseteq \mathcal{L}$:

$$\boldsymbol{\pi}_0 \triangleq \mathcal{S}$$

$$\boldsymbol{\pi}_{k+1} = \mathcal{P}\boldsymbol{\pi}_k \cap \boldsymbol{\pi}_k.$$

The greater is k the tighter is the polytope $\boldsymbol{\pi}_k$, and any $\boldsymbol{\pi}_k \supseteq \mathcal{L}$.

In both cases – uncertainties that affect directly the steady-state probabilities $\overline{\pi_\tau}$ or the transition probability matrix P – the synthesis of the robust conditioning Markov chain can then be formulated as the following Linear Programming problem:

$$\begin{array}{ll} \text{Find a vector } \overline{\pi}_\sigma = \begin{bmatrix} \overline{\pi}_{\sigma_1} & \cdots & \overline{\pi}_{\sigma_n} \end{bmatrix}^T \text{ such that} & \\ 1) \quad (M_c M_{\pi_\tau})^T \overline{\pi}_\sigma < 0 & \\ 2) \quad 0 < \overline{\pi}_{\sigma_i} < 1 & \\ 3) \quad \sum_{i=1}^{n} \overline{\pi}_{\sigma_i} = 1, & \end{array} \quad (4)$$

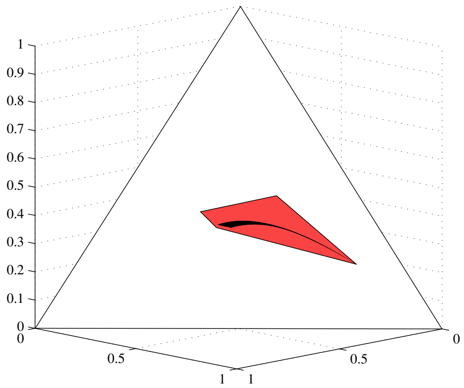


Fig. 3. Example of polytopic approximation $\boldsymbol{\pi}$ for the uncertain set $\mathcal{L}$. The simplex $\mathcal{S}$ is also depicted.

where

$$M_{\pi_\tau} = \begin{bmatrix} \pi_\tau^1 & \pi_\tau^2 & \cdots & \pi_\tau^r \end{bmatrix}$$

and r is the number of vertices of the polytopic approximation of the uncertain steady-state scheduler probabilities.

Example 1: Let us consider the following stochastic matrices as vertices of the matrix polytope $\mathcal{P}$

$$P_1 = \begin{bmatrix} 0.36 & 0.28 & 0.36 \\ 0.43 & 0.16 & 0.41 \\ 0.38 & 0.41 & 0.20 \end{bmatrix}, \quad P_2 = \begin{bmatrix} 0.59 & 0.15 & 0.26 \\ 0.35 & 0.33 & 0.32 \\ 0.11 & 0.52 & 0.37 \end{bmatrix},$$

$$P_3 = \begin{bmatrix} 0.29 & 0.09 & 0.62 \\ 0.06 & 0.86 & 0.08 \\ 0.19 & 0.43 & 0.38 \end{bmatrix}.$$

The set of allowable probability distribution vectors $\mathcal{S}$ is given by the simplex having as vertices the canonical vectors e_1^T, e_2^T and e_3^T (see fig. 3). In the same figure the non convex set $\mathcal{L}$ is represented by means of some points randomly extracted by it. After 10 iterations of the previous algorithm, the polytopic approximation

$$\begin{aligned} \boldsymbol{\pi}_{10} &= \text{conv}\{\pi_\tau^1, \dots, \pi_\tau^6\} \\ \pi_\tau^1 &= \begin{bmatrix} 0.2024 & 0.3851 & 0.4125 \end{bmatrix}^T \\ \pi_\tau^2 &= \begin{bmatrix} 0.4009 & 0.2805 & 0.3186 \end{bmatrix}^T \\ \pi_\tau^3 &= \begin{bmatrix} 0.3993 & 0.2875 & 0.3132 \end{bmatrix}^T \\ \pi_\tau^4 &= \begin{bmatrix} 0.4142 & 0.2233 & 0.3625 \end{bmatrix}^T \\ \pi_\tau^5 &= \begin{bmatrix} 0.1116 & 0.6883 & 0.2001 \end{bmatrix}^T \\ \pi_\tau^6 &= \begin{bmatrix} 0.1926 & 0.4179 & 0.3895 \end{bmatrix}^T \end{aligned}$$

for $\mathcal{L}$ is obtained. $\boldsymbol{\pi}_{10}$ is depicted in fig. 3 as the red polytope enclosing the set of points representing $\mathcal{L}$.

V. Examples

The control of a Furuta pendulum with zero offset ([19]), depicted in fig. 4, will be used to illustrate the application of

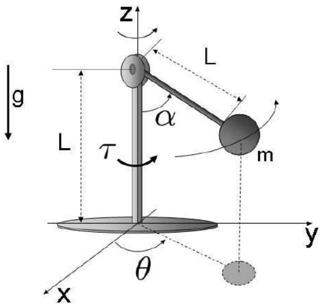


Fig. 4. Mechanical system adopted for the anytime controller simulations: a Furuta pendulum with zero offset ([19]).

the robust switching policy. The nonlinear dynamic equations are

$$ml^2\ddot{\alpha} - \frac{1}{2}ml^2 \sin 2\alpha\ \omega^2 + mgl \sin \alpha = 0$$
$$(I + ml^2 \sin^2 \alpha)\dot{\omega} + ml^2 \sin 2\alpha\ \dot{\alpha} = \tau,$$

where $m = 1$ kg is the hanging mass, $l = 1$ m is total length of the rigid vertical bar and of the rigid bar that hangs the mass, $I = 10^{-3}$ kg/m^2 is the inertia of the rotating vertical bar and $g = 9.8$ m/s^2 is the gravity acceleration. Let α be the measured orientation angle between the rigid vertical bar and the hanging bar and ω be the angular velocity of the vertical bar. The Furuta pendulum is actuated by the torque control τ applied at the structure basement. Let $\mathbf{x} = [x_1, x_2, x_3]^T = [\alpha, \omega, \dot{\alpha}]^T$ be the state space vector. Consider the linearized system with respect to the equilibrium point $\bar{\mathbf{x}} = \left[\pi/4, \sqrt{\frac{g\sqrt{2}}{l}}, 0\right]^T$, with $\tau = 0$. Sampling the linearized system with an adequate sample time, the open loop unstable, discretized transfer function of the system will be

$$G(z) = \frac{0.0012178(z + 3.657)(z + 0.2734)}{(z - 1)(z^2 - 1.664z + 1)}.$$

Three controllers are designed

$$C_1(z) = \frac{31.6(z^2 - 1.8z + 1.1)}{(z - 0.1)(z - 0.5)}$$
$$C_2(z) = \frac{117.7(z - 1.3)(z - 4.7\,10^{-3})(z^2 - 1.4z + 0.7)}{(z - 0.5)(z + 0.2)(z - 0.1)(z^2 + 0.4z + 0.9)}$$
$$C_3(z) = \frac{1370.9(z - 0.4)(z - 0.6)(z - 0.2)(z + 0.2)}{(z - 0.5)^2(z + 0.3)^2(z - 0.1)^2} \times \frac{(z - 4.7\,10^{-3})(z^2 - 0.7z + 0.2)(z^2 - 0.4z + 0.3)}{(z^2 + 0.6z + 0.8)(z^2 + 3.4z + 4.7)}.$$

The controller $C_1(z)$ is designed to ensure the stability requirement, while $C_2(z)$ and $C_3(z)$ are obtained applying twice in cascade an LQG design technique, thus resulting in a quite large number of states for a state space realization (see [4] for a detailed discussion on the hierarchical controllers implementation for the anytime controllers). Prefilters $F_i(z)$ for the reference signal (see figure 2) are assumed to be constants mainly used to adapt the steady-state gain and ensure static requirements, whose values are

$$F_1(z) = 21.28, F_2(z) = 0.54 \text{ and } F_3(z) = 2.73.$$

Assuming a Markov description for the scheduler, the nominal steady-state probability distribution is roughly known to be $\overline{\pi}_\tau = [1/20, 5/20, 14/20]$, with a transition probability matrix

$$P_\tau = \begin{bmatrix} 0.2744 & 0.342 & 0.3836 \\ 0.0881 & 0.3443 & 0.5676 \\ 0.0204 & 0.2097 & 0.7699 \end{bmatrix}.$$

Assuming the nominal scheduler stochastic description, solving the Linear Programming problem 3 leads to a steady-state conditioning probability distribution $\overline{\pi}_\sigma = [0.017, 0.98, 0.003]$. The resulting conditioned distribution $\overline{\pi}_d = [0.058, 0.94, 0.002]$ thus satisfies the 1-step average contractivity condition (2), hence AS-stability is guaranteed.

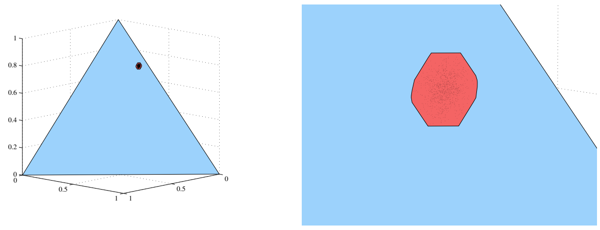
Let us now consider an uncertainty of the scheduler Markov chain transition probability matrix P_τ of the kind

$$\tilde{P}_\tau = P_\tau + \Delta P_\tau = P_\tau + \begin{bmatrix} \rho_{11} & \rho_{12} & \rho_{13} \\ \rho_{21} & \rho_{22} & \rho_{23} \\ \rho_{31} & \rho_{32} & \rho_{33} \end{bmatrix},$$

where ρ_{ij} are uncertainties such that $\rho_{1j} \in [-0.025, 0.025]$, $\rho_{2j} \in [-0.02, 0.02]$ and $\rho_{3j} \in [-0.01, 0.01]$. It is worth noting that $\tilde{P}_\tau$ may be not a stochastic matrix, since ΔP_τ is a "box shaped" uncertainty. The valid matrix polytope is the result of the intersection of the set of the perturbed matrices $\tilde{P}_\tau$ with the set of stochastic matrices. This operation can be easily performed as the intersection of two polytopes of suitable dimensions. The matrix polytope $\mathcal{P}$ for this example has 216 vertices and it is not explicitly reported. Using the algorithm presented in Section IV, an approximating polytope of the steady-state probability set $\mathcal{L}$ is obtained, with 30 vertices. The figure 5-a depicts the uncertain steady-state probabilities of the scheduler $\boldsymbol{\pi}_8 = \text{conv}\left\{\pi_\tau^1, \ldots, \pi_\tau^{30}\right\}$. In figure 5-b, the polytopic approximation is magnified together with with randomly chosen points from the set $\mathcal{L}$.

Solving the Linear Programming problem 4, a polytope $\mathcal{C}$ of conditioned steady-state probabilities and the stochastic description of the conditioning Markov chain $\overline{\pi}_\sigma = [0.0003, 0.9996, 0.0001]$ are obtained.

In fig. 6, the *Root Mean Squares* (RMS) of the regulation error for different closed loop controllers is shown, corresponding to perturbed initial conditions $x_0 = [0, \pi/10, 0]^T$.



a) b)

Fig. 5. Convex set of the steady-state probabilities generated by the uncertainties on the transition probability matrix P_τ.

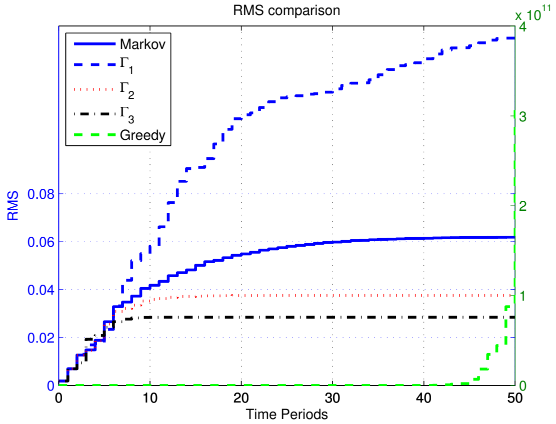


Fig. 6. Regulation results for the Furuta pendulum example: mean value of the RMS errors (for one thousand runs) of the closed loop system with different control schedules.

The RMS regulation error is obtained as the mean of the RMS errors for one thousand simulations, whereby the scheduler stochastic description is obtained by randomly choose a transition probability matrix $\tilde{P}_{tau} \in \mathcal{P}$. Plots labelled Γ_1, Γ_2, and Γ_3, corresponding to results obtained without switching, are reported for reference (due to the bottom-up design, the performance increase using more complex controllers). Furthermore, it is worthwhile to note that the RMS obtained by the greedy switching policy, i.e. the policy that executes the maximum scheduled controller, shows instability: the axis labels on the right apply to this plot. On the same figure 6, the plot labelled “Markov” shows the RMS errors mean obtained by the stochastically conditioned scheduler.

The example shows how the proposed stochastic switching policy ensures the AS-stability of the closed loop system (which is not guaranteed by the greedy policy), while it obtains a definite performance increase (of the order of 50%) with respect to the conservative scheduler (corresponding to using only Controller 1, see fig. 6) even in the presence of a stochastic uncertain description of the scheduler.

VI. Conclusions

We considered the problem of robustly scheduling the execution of different, hierarchically ordered tasks designed for anytime control of a linear plant. Given an uncertain stochastic model of the scheduler, and the set of controllers, we are able to provide a switching policy that conditions the partially known scheduler so that the resulting switching system is stable in a probabilistic sense.

Firstly, we presented an algorithm that approximates the non convex set of the steady-state probabilities of the uncertain Markov chain modelling the scheduler. Then, we derived a linear program that finds the robust switching policy under the estimated scheduler uncertainties. We have also shown that underexploitation of CPU time caused by conservative control scheduling policies can be effectively reduced, and control performance can be enhanced by adopting a robust scheduling policy previously designed.

Much work remains to be done to make numerically tractable the solution in the robust multi–step case.

References

[1] C. L. Liu and J. W. Layland, “Scheduling algorithms for multiprogramming in a hard-real-time environment,” *Journal of the Association for Computing Machinery*, vol. 20, no. 1, 1973.

[2] J. W. S. Liu, *Real–Time Systems*, T. Holm, Ed. Upper Saddle River, NJ: Prentice Hall Inc., 2000.

[3] G. Buttazzo, *Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications*. Boston: Kluwer Academic Publishers, 1997.

[4] D. Fontanelli, L. Greco, and A. Bicchi, “Anytime control algorithms for embedded real-time systems,” in *Proc. of Hybrid Systems: Computation and Control*, St. Louis, MO, April 2008.

[5] J. W. S. Liu, W.-K. Shih, K.-J. Lin, R. Bettati, and J.-Y. Chung, “Imprecise computation,” *Proceedings of the IEEE*, vol. 82, no. 1, pp. 83–93, January 1994.

[6] N. Perrin and B. Ferri, “Digital filters with adaptive length for real–time applications,” in *Proc. IEEE Real-Time and Embedded Technology and Applications Symposium*, K. E. Le Royal Meridien, Ed., Toronto, Canada, May 2004.

[7] M. S. Branicky, “Stability of hybrid systems: State of the art,” in *Proc. 36th IEEE Conf. On Decision and Control*, San Diego, California, USA, December 1997, pp. 120–125.

[8] R. A. DeCarlo, M. S. Branicky, S. Pettersson, and B. Lennartson, “Perspectives and results on the stability and stabilizability of hybrid systems,” *IEEE Proceedings*, vol. 88, no. 7, pp. 1069–1082, 2000.

[9] H. Ye, A. N. Michel, and L. Hou, “Stability theory for hybrid dynamical systems,” *IEEE Trans. Automat. Contr.*, vol. 43, no. 4, pp. 461–474, 1998.

[10] J. P. Hespanha and A. S. Morse, “Switching between stabilizing controllers,” *Automatica*, vol. 38, no. 11, pp. 1905–1917, 2002.

[11] M. Wicks and R. DeCarlo, “Solution of coupled Lyapunov equations for the stabilization of multimodal linear systems,” in *Proc. American Control Conf.*, Albuquerque, NM, June 1997, pp. 1709–1713.

[12] S. Pettersson and B. Lennartson, “Stabilization of hybrid systems using a min-projection strategy,” in *Proc. American Control Conf.*, Arlington, Virginia, June 2001, pp. 223–228.

[13] X. D. Koutsoukos and P. J. Antsaklis, “Design of stabilizing switching control laws for discrete- and contiunuous-time linear systems using piecewise-linear Lyapunov functions,” *Int. J. Control*, vol. 75, no. 12, pp. 932–945, 2002.

[14] L. Greco, D. Fontanelli, and A. Bicchi, “Almost sure stability of anytime controllers via stochastic scheduling,” in *Proc. IEEE Int. Conf. on Decision and Control*, New Orleans, LO, December 2007.

[15] Y. Fang, K. Loparo, and X. Feng, “Almost sure and δ-moment stability of jump linear systems,” *Int. J. Control*, vol. 59, no. 5, pp. 1281–1307, 1994.

[16] P. Bolzern, P. Colaneri, and G. D. Nicolao, “On almost sure stability of discrete-time Markov jump linear systems,” in *Proc. 43rd IEEE Conf. On Decision and Control*, vol. 3, 2004, pp. 3204–3208.

[17] R. K. Brayton and C. H. Tong, “Stability of dynamical systems: A constructive approach,” *IEEE Trans. Circuits Syst.*, vol. CAS-26, no. 4, pp. 224–234, 1979.

[18] ——, “Constructive stability and asymptotic stability of dynamical systems,” *IEEE Trans. Circuits Syst.*, vol. CAS-27, no. 11, pp. 1121–1130, 1980.

[19] K. Furuta, M. Yamakita, and S. Kobayashi, “Swing-up control of inverted pendulum using pseudo-state feedback,” *Proceedings of the Institution of Mechanical Engineers. Pt.I. Journal of Systems and Control Engineering*, vol. 206, no. I4, pp. 263–269, 1992.

[20] A. Cervin, B. Lincoln, J. Eker, K.-E. Årzén, and G. Buttazzo, “The jitter margin and its application in the design of real-time control systems,” in *Proc. 10th Int. Conf. on Real-Time and Embedded Computing Systems and Applications*, Gothenburg, Sweden, August 2004.

[21] D. Liberzon, J. P. Hespanha, and A. S. Morse, “Stability of switched systems: A Lie-algebraic condition,” *Systems & Control Letters*, vol. 37, no. 3, pp. 117–122, 1999.